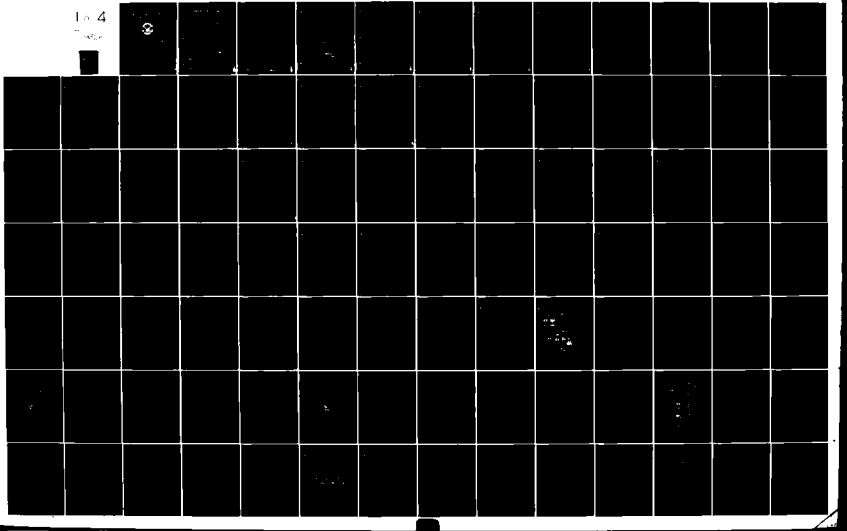
AD-A119 326 NAVAL POSTGRADUATE SCHOOL MONTEREY CA F/G 15/7
THE ENGINEER EFFECTS MODULE FOR THE STAR COMBAT MODEL.(U)
MAR 82 S C MAIN, J V MUDD

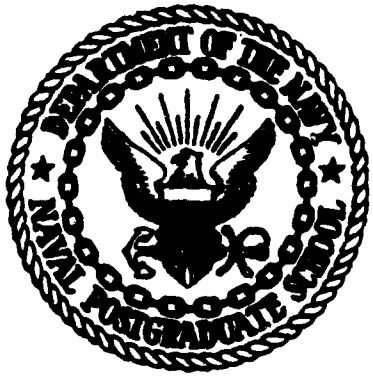
UNCLASSIFIED NL

1 of 4



(2)

NAVAL POSTGRADUATE SCHOOL
Monterey, California



THESIS

THE ENGINEER EFFECTS MODULE FOR THE STAR COMBAT MODEL

by

Stephen Charles Main

and

James Vincent Mudd

March 1982

Thesis Advisor: James K. Hartman

Approved for public release: distribution unlimited

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A122326	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) The Engineer Effects Module for the STAR Combat Model	5. TYPE OF REPORT & PERIOD COVERED Master's Thesis March 1982
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Stephen Charles Main James Vincent Mudd	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940	12. REPORT DATE March 1982 13. NUMBER OF PAGES 379
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release: distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

DTIC ELECTE SEP 17 1982 S A

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Minefield STAR

Army Obstacles

Combat Simulation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This thesis presents an Engineer Effects Module for the Simulation of Tactical Alternative Responses (STAR) combat model. The effects of engineer obstacles on the combat process are important, and the STAR combat model previously lacked the capability to model these effects. Implicit in the construction of the model is the task of modelling the obstacles followed by the simulation of the synergistic effects of the obstacles on the

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE S/N 0102-014-6601

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

combat commander's actions. The Engineer Effects Module is executed in the SIMSCRIPT II.5 programming language. This thesis serves as both the implementation and running instructions for the Engineer Effects Module for the STAR model.

Approved for public release: distribution unlimited

The Engineer Effects Module for the STAR Combat Model

by

Stephen C. Main
Captain, United States Army
B.S., United States Military Academy, 1972

and

James V. Mudd
Captain, United States Army
B.S., United States Military Academy, 1974

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
March, 1982

Authors: Stephen C. Main

James V. Mudd

Approved by: James K. Hartman
Thesis Advisor

S. H. Parry
Second Reader

K. T. Marshall
Chairman, Department of Operations Research

W. M. Woods
Dean of Information and Policy Sciences

ABSTRACT

This thesis presents an Engineer Effects Module for the Simulation of Tactical Alternative Responses (STAR) combat model. The effects of engineer obstacles on the combat process are important, and the STAR combat model previously lacked the capability to model these effects. Implicit in the construction of the model is the task of modelling the obstacles followed by the simulation of the the synergistic effects of the obstacles on the combat commander's actions. The Engineer Effects Module is executed in the SIMSCRIPT II.5 programming language. This thesis serves as both the implementation and running instructions for the Engineer Effects Module for the STAR model.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

I. INTRODUCTION

The Simulation of Tactical Alternative Responses (STAR) combat model, as it currently exists, does not model Engineer obstacles or counter-obstacle operations. It is the purpose of the Engineer Effects Module to install this capability in the STAR Combat model. This Engineer Effects Module consists of the representation of four types of obstacles and their effects on the battlefield processes. The obstacles are the minefield, tank ditch, road crater, and the destroyed bridge (short-wet gap). The minefield actually consists of the modelling of five different types of fields, the anti-tank patterned minefield, the anti-tank scatterable minefield, the anti-personnel blast minefield, the anti-personnel fragmentation minefield and the Claymore mine.

The obstacles are important in their own right, however, the synergistic effects of the obstacles on the forces and tactics being employed are the crucial effects to be modelled. The modelling of land combat and the ability to capture synergistic effects on the battlefield are difficult and complex tasks.

In order to model the effects of the Engineer obstacles mentioned above, the counter-obstacle tactics for the combat forces had to be developed and modelled. The tactics that are modelled for the the obstacles are as follows:

1. For the minefield:

   Conduct a hasty breach with plows.

   Conduct a bypass.

   Bull through the field.

2. For tank ditches and road craters:

   Conduct a crossing of the gap with an Armored Vehicle Launched Bridge

   Conduct a lane filling operation using tanks equipped with dozer blades.

   Conduct a self-breaching operation by allowing each vehicle to enter the obstacle and create a breach by caving the obstacle walls in.

   Conduct a bypass of the obstacle, when no proper equipment is present and the risk of trying a self breach is too great.

3. For the short-wet gap

   Conduct a hasty crossing using the Armored Vehicle Launched Bridge (AVLB).

   Conduct a bypass of the gap if there is no AVLB.

The capability to breach obstacles necessitated the introduction of a new system in STAR, the armored vehicle launched bridge (AVLB), and the expansion of the capability of existing systems through the representation of mine plows and bull dozer blades.

In Chapter 2, the current capabilities and equipment of the United States Army are discussed. This information forms the basis for the effects that are modelled for the United States and NATO forces.

In Chapter 3, the capabilities and equipment of the forces of the Soviet Union are described and discussed. This chapter forms the basis for the tactics that are modelled for the Warsaw Pact forces in STAR.

Chapter 4 consists of a review of some of the past modelling efforts and ideas that exist in the Engineer arena.

Chapter 5 consists of a detailed description of the minefields and their associated tactical responses constructed for the STAR combat model.

Chapter 6 is the detailed description of the remaining obstacles (tank ditch, road crater, and destroyed bridge) and their synergistic effects.

Finally, Chapter 7 consists of some enhancements the authors see as possibilities for the model in the future. Modeling is viewed by the authors as an iterative process and this chapter may assist in the future development of the model. In the appendices, variables are defined, changes to

existing STAR code are explained, and the computer code is listed and described in detail. Complete running instructions are also included.

It is the intention of the authors that this Engineer effects module can be fully implemented by a serious user of the STAR combat model using only this document and the existing STAR documentation.

II. UNITED STATES TACTICS AND EQUIPMENT

History has shown that obstacles and barrier systems are an effective casualty producing ingredient on the battlefield. Most obstacles used were minefields or obstacles reinforced with mines. The following table lists Allied tank losses to mines as a percentage of losses from all enemy action [Ref. 1].

THEATER	PERCENT
North Africa 1942-43	18
Western Europe 1944-45	23
Italy 1943-45	28
Pacific 1944-45	34
Korea 1950-55	56
Vietnam 1967-69	69

More recently, obstacle use played a major role in the Mid-East War of October 1973. On the plains of the Golan Heights, the Israelis constructed an anti-tank ditch approximately 15 kilometers long. The ditch was heavily reinforced with minefields on both the enemy and the friendly sides. The combination of the Israelis' decisive use of the terrain/terrain reinforcement and mines with covering fire allowed their grossly outnumbered force of 180 tanks, 11 batteries of Artillery, and supporting Engineers to defeat an attacking Syrian force of 800 tanks, 115

batteries of Artillery, and Engineers [Ref. 2]. The above historical evidence highlights the previous use of obstacles on battlefields of the past. The evidence should spark an interest in how their use will affect the battlefield of today and tomorrow.

The STAR model, as it currently exists, does not model any Engineer obstacles. STAR does however, possess the inherent capability to do so using the field module with some major modifications [Ref. 3]. This chapter will briefly describe the the following items as a prelude to attempting to model Engineer effects in STAR.

1. Four general obstacles:

   the minefield

   the road crater

   the tank ditch

   the destroyed bridge (short-wet gap)

2. The alternatives available to the tactical commander in order to overcome an obstacle in his path of advance.

In this model, the actual employment of the Engineers and their equipment will not be discussed. If the reader wishes to investigate this area more deeply, he should refer to References 4, 5, and 6. It is acknowledged that the

combat Engineer is a viable member of the combined arms team. Engineers increase the combat effectiveness of the combat forces on the battlefield. However, in order to model the synergistic effects of Engineer obstacles in STAR, the existing model must form the basis of the iterative process. After the obstacles and their effects are modelled, then the Engineer units may be added to the battle being played. In short, the first step in the process is to install an Engineer Obstacle Effects Module in STAR.

A. OBSTACLES

An obstacle is any obstruction that stops, delays, or restricts movement. Obstacles have been classified as natural/existing or artificial/reinforcing/manmade. The primary purposes of obstacles are to enhance the effectiveness of friendly fires, to delay and disrupt the enemy formations, to divert the enemy, to allow the tactical commander to use economy of force and to enable him to protect his flanks. Because threat tactics are based on the massive use of mechanized forces, one could say that the principal purpose of battlefield obstacles is to overcome the inherent mobility advantage of the enemy's tank forces [Ref. 7]. For the purpose of this discussion, only the four

major anti-armor obstacles previously mentioned will be discussed.

1. The Minefield

Minefields are located where they will delay armor and strengthen the defense, but they are not used with the idea of blocking or destroying the enemy force. Mines reinforce natural obstacles, scare the enemy, cause his attention to be diverted from the defender, and influence his maneuver. [Ref. 8] For the purpose of model building, the names of the various types of minefields are not important but the methods of installation are. There are four basic techniques used in laying minefields: [Ref. 9]

a. Hasty Minelaying

Hasty minelaying is done near the forward edge of the battlefield without Engineer support. Mines are emplaced so that friendly troops may detect and recover the mines. Usually, they are laid on top of the ground in a random fashion or in a specific pattern, i.e. a row.

b. Deliberate Minelaying

Deliberate minelaying normally follows a pattern. Each mine is buried and camouflaged. Mine clusters and strips are plotted and marked. Boundaries are fenced

and marked. This type of minelaying is normally done by the Engineers and is used when there is time to plan, organize, and prepare the logistical support for the effort.

c. Special Minelaying

Special minelaying is the mining of railroads, routes, stream beds, and other unique point targets on the battlefield. It is normal to improvise when laying mines for these purposes.

d. Scatterable Minelaying

Mines can be delivered by artillery, aircraft, missiles, ground dispensers, or thrown by hand. This method allows the commander to have a rapid minefield installation capability that is not manpower intensive.

The other three categories of obstacles can be thought of as anti-vehicular obstacles. Anti-vehicular obstacles should not be continuous across the front of a position, but should have gaps which can be kept under observation and covered by fire. Such gaps tend to channelize the enemy's movement. These obstacles are located to take advantage of natural concealment for a surprise effect and are usually reinforced with mines. [Ref. 10]

2. The Tank Ditch

The tank ditch is a linear obstacle emplaced normally by mechanical digging machines. There are two primary types of ditches, the triangular ditch and the rectangular ditch. The triangular ditch has a 1.5 to 1.8 meter vertical wall with berm and the rectangular ditch is 3.3 meters wide, 1.5 to 1.8 meters deep and also has a berm. The vertical wall or depth varies because of different soil types. The triangular ditch does not present an obstacle to the friendly or counter-attacking force. [Ref. 2] The rectangular ditch presents an obstacle to both the attacking force and the friendly force.

3. The Road Crater

Crater type obstacles are used for blocking roads, trails, or defiles. The location of the crater is chosen such that no bypass exists or if one exists, the bypass route can either be mined, covered by friendly fire or both. Craters are formed by explosive charges installed and prepared in advance for later detonation. [Ref. 10] Road Craters, in order to be effective obstacles, must be too wide to be spanned by the natural bridging capability of tracked vehicles. In addition, they must be deep and steep-

sided to prevent vehicles from passing through them. Road craters created by blast will not stop modern tanks indefinitely, but are effective obstacles if the tank requires three or more passes in order to cross the crater. Three passes provides sufficient time for the anti-tank weapon to destroy a tank by fire. [Ref. 11]

4. The Destroyed Bridge (short-wet gap)

Generally, bridges are destroyed to create obstacles which delay the enemy. However, bridges seldom require complete destruction. The method used for demolition should normally permit the economical reconstruction of the bridge by friendly troops in future operations [Ref. 11].

Now that the four basic obstacles have been described, one may ask the question, What makes an obstacle effective? An effective obstacle is one which slows and delays the enemy and is covered by fire. This is to counter enemy efforts to breach and/or destroy the enemy while entrapped by the obstacle. Obstacles should be employed in such a way as to surprise the enemy and take advantage of his confusion. Finally, they should be of no advantage to the enemy, any cover and concealment provided by the obstacle should be mined or booby-trapped. [Ref. 8]

B. TACTICAL RESPONSES TO OBSTACLES

The enemy force has the same capabilities and opportunities to emplace obstacles as the friendly force. Therefore, it is important to discuss the friendly tactical commander's possible actions upon encountering an obstacle on the battlefield. The friendly commander must keep one principle in mind when an obstacle is encountered, maintain the momentum of the offense. Obstacles must not impede the movement for unusually long periods of time. The commander has three alternatives when faced with an obstacle; he may bypass, "force through" or breach the obstacle. [Ref. 7]

1. The Bypass

To conserve time and manpower, obstacles are bypassed whenever possible. However, if the enemy has employed these obstacles cleverly, they will be difficult to bypass [Ref. 7] . The defender has probably covered all bypass routes by fire. The unit should conduct the bypass with attention to cover, concealment, and suppression. [Ref. 12]

2. The Breach

If bypassing is not possible, then another choice for the commander is to attempt a breach. A breach is

conducted when the unit possesses the proper equipment to breach. Two methods of breaching may be employed, the assault breach or the deliberate breach. [Ref. 7]

a. The assault/hasty breach is done quickly during either hasty or deliberate attacks. This is a tactical breach. The assault troops are under enemy direct or indirect fire. The main objective is speed in gaining the breach. Delays while breaching, are more costly in terms of causing casualties than the mines.

b. The deliberate breach is conducted when enemy fires covering the field have been neutralized and safety is more important than time. The performance of a deliberate breach is an Engineer responsibility. Because Engineer units are not currently modelled in STAR, this tactic will not be discussed as an alternative for the tactical commander.

3. <u>The Force Through</u>

The "force/bull through" is attempted when no other way to overcome the obstacle is possible. The unit does not possess any breaching equipment and will just drive their vehicles through the minefield hoping to get through and perhaps clear a path. Heavy losses are expected when this tactic is employed. [Ref. 12]

The breach options demand further explanation. The tactical commander should breach obstacles by applying these fundamentals: [Ref. 7]

1. Suppress enemy weapons using available fire.
2. Use smoke to obscure enemy observation.
3. Secure the far side of the obstacle.
4. Reduce the obstacle.

The fourth fundamental is of specific interest in order to model obstacles in STAR. How does the tactical commander reduce obstacles, or to be more specific, how does he combat the obstacles discussed in Section A of this chapter?

a. Neutralizing and Breaching Minefields

The use of mechanical or explosive breaching devices is preferred over manual methods unless a special need exists. [Ref. 7] Explosive devices require Engineer support and for this reason are not discussed. The tactical commander therefore has only a tank mounted dozer blade (one per tank company) that can breach small point minefields or the Europe-deployed tank mounted mine roller for larger minefields [Ref. 6].

b. Tank Ditches/Craters

A tank dozer can be used to push down the sides of ditches or to fill craters in. If the gap created is 57 feet or less then the AVLB can be used. [Ref. 12] It should be noted that tank ditches and road craters are usually reinforced with mines and may require additional breaching equipment.

c. Crossing Short-Wet Gaps

Short gaps created by streams, small rivers, or valleys can be crossed using the existing civilian fixed bridge structures. However, if the enemy has destroyed these, or there were none in existence, then the tactical commander must resort to his organic assets. Normally, the AVLBs are utilized to cross gaps of less than 57 feet (17.4 meters) with vehicles of up to 60 tons in weight [Ref. 6].

This concludes the discussion of obstacles and tactical responses available to a commander of U.S. forces. The tactics and capabilities described in this chapter form the basis for modelling Engineer effects on Blue forces currently in the STAR model. The next chapter will discuss the capabilities and tactical alternatives of the Soviet Army.

III. SOVIET TACTICS AND EQUIPMENT

A. INTRODUCTION

This chapter is a consolidation of information on the Soviet forces and their capabilities in the areas of counter-obstacle operations and equipment. All the information presented here is unclassified and available in open literature. It is a widely accepted fact that Soviet military operations and tactics are directed towards the offensive. It is for this reason, that Soviet defensive operations will not be discussed. The primary mission of the Soviet Engineer is to enhance the mobility of the Soviet ground combat forces by countering obstacles to movement, be they natural or manmade. To construct an Engineer Effects Module for the Simulation of Tactical Alternative Responses (STAR) combat model one must determine the tactics for the Red forces. The research for this chapter provided the bulk of the tactical and equipment information required for the implementation of an engineer effects module for STAR.

The Soviet Union is preparing itself for a massive Air-Land battle with the forces of NATO. It has long been

hypothesized that the Warsaw Pact, led by the forces of the Soviet Union, will eventually attack west across the borders of Czechoslovakia and the German Democratic Republic into Western Europe. In order to be successful in their attack, the Warsaw Pact forces will rely on massive combined arms forces attacking on several avenues of advance with great speed. The NATO forces, on the other hand, will be defending with smaller forces. It is for this reason that the NATO forces will rely heavily on terrain reinforcement to slow the advancing Pact forces. These reinforcements will include minefields, tank ditches, road craters, blown bridges and other engineer obstacles. The Soviet Union is aware of this and consequently has amassed a rather impressive array of counter-obstacle forces and equipment. Further, in the Great Patriotic War the Soviet Union used many minefields and obstacles as terrain reinforcements against the advancing forces of Nazi Germany. [Ref. 13] In that war, the Soviet Union gave Nazi Germany a lesson on the effects of obstacles on the movement of ground combat forces. That lesson has not been forgotten by the teacher. Most of the high Soviet leadership of today fought in that war and thus, remembers the experience. The Soviet Union

not only is preparing to attack through the NATO defenses, but believes that it can be done successfully.

The Soviets developed into first-rate users of mine and obstacle warfare during the Great Patriotic War. It has been said that the battle of Kursk is the finest example of mine warfare in history. In that battle, some 3 million mines were laid to stop the Germans. In order to avert a transportation problem caused by the heavy mine usage, every time a soldier was sent to the front, he was required to take two mines with him. [Ref. 13, 14] The lessons learned as great users of obstacles have helped the Soviet Union to develop an arsenal of counter-obstacle equipment unparalleled in the world today.

It is interesting to note that the Soviet Union and the Warsaw Pact claim to be defensive forces, yet possess an offensive capability that is in a word, awesome. The NATO forces, though claimed to be the potential aggressors, possess a counter-obstacle capability that pales in comparison to that of the Warsaw Pact. As an example, the total mine roller capability of the US Army in Europe consists of 10 rollers that were scheduled for delivery to USAREUR in late 1981 [Ref 15]. The Soviet forces possess 9

rollers in just one Engineer Company supporting a hasty minefield breach of the forward combat elements [Ref. 13].

B. SOVIET FORCES AND EQUIPMENT

The Soviet Union possesses an impressive array of counter-obstacle equipment and forces in order to enhance the mobility of the Soviet combat arms. This description will focus on two primary areas:

1. The organic counter-obstacle elements in the ground combat arms.
2. The Engineer units organic to the larger units.

Perhaps the most obvious and least mentioned item is the design characteristics of the Soviet armored vehicles. Most of these vehicles are amphibious so that many water obstacles may be overcome easily with no additional bridging support. Although bridging equipment may be desirable, in many cases it is faster to just drive up to the river bank, enter the water, and cross. Most of the tanks in the Soviet army have an inherent snorkeling capability that allows them to cross water obstacles as deep as 5.5 meters at speeds of 2 km/hr [Ref. 15]. Table 1 is a consolidated list of the amphibious capability of some Soviet vehicles.

Table 1: Soviet Army Amphibious Vehicle Capabilities

Vehicle/system	Water speed	Additional attributes
BTR 50/60	10 km/hr	
BMP	6 km/hr	can fire 73 mm main gun and Sagger while swimming
PT 76	10 km/hr	can fire 76mm main gun while swimming
BRDM	10 km/hr	
122mm SP HOW	approx 10 km/hr	
SA-8/SA-9	approx 10 km/hr	

[Ref. 16,17]

The above organic counter-obstacle capability is further supplemented by bridging assets at the Regimental level. These bridges can be used to cross water obstacles with no bridges left intact and may also be used to counter tank ditches and road craters. The following is a consolidation of the assets organic to the Motorized Rifle Regiment (MRR) and the Tank Regiment (TR).

Table 2: Organic Breaching Equipment of a Soviet Regiment

Vehicle/system	Basis of issue	Capability
MTU-20 (tank mtd bridge)	3 per Regt	12.3 to 20 meters
TMM (truck launched br)	4 per Regt	10.5 meters ea or 1 42 meters long
BTU Dozer Blades	3 per Regt	100-300 meters per hr by 3.4 meters wide

[Ref. 16,17,18]

In addition to these capabilities at the Regimental level, other assets exist in the Divisional Engineer Battalion to aid in counter-obstacle operations. They include the following:

Table 3: Organic Breaching Equipment of a Soviet Division

Vehicle/system	Basis of issue	Capability
BTU Dozer Blades	3 per div bn	100-300 meters per hr by 3.4 meters wide
MTU-20	4 per div bn	12.3 to 20 meters
THM	8 per div bn	10.5 meters ea or combined together.
PTS(tracked amphibian)	14 per div bn	carries 15000 kg of cargo at 15 km/hr
GSP(Ferry)	8 per div bn MRD 12 per div bn TD	4 50 metric ton ferries 6 50 metric ton ferries
PMP(pontoon bridge)	18 per div bn	119 meters of 60 ton capacity bridge

[Ref. 16,17,18]

The Front and Army levels each have additional quantities of primarily the same equipment already mentioned. Table 4 lists the additional quantities at the Front or Army levels.

The man-made obstacles that will undoubtedly pose the greatest problems for the Soviet Commander are minefields. At the unit level, the mine-plow will be the primary asset, and it will be available in great numbers. The estimates vary, however it appears that the minimum number will be one

Table 4: Organic Breaching Equipment of a Soviet Front/Army

Vehicle/system	Quantiity	Capability
BTU Dozer Blades	9	100-300 meters per hr by 3.4 meters wide
MTU-20	24	12.3-20 meters
TMM	48	10.5 meter ea
PMP	108	681 meters of 60 ton capacity bridge
PTS	98	15000 kg ea cargo
GSP	48	up to 110 tons at 7-8 km/hr
		[Ref. 16,17,18]

plow per platoon, i.e. at least one tank in three will be equipped with a mine plow. However, normally each forward platoon has two tanks fitted with mine plows. [Ref. 19] Every T-72 has the capability to mount a mine plow. [Ref. 16] The following is a table of the mine clearing equipment mounted on tanks:

Table 5: Tank-mounted Mine Clearing Equipment

System	Basis of issue	Capability
KMT4/6 Plow	3 per tank Co	10 km/hr 35 cm deep
PT54-M Roller	3 per Regt and 3 per Eng Bn	10 km/hr
KMT-5 Plow/Roller Combination	4 per Regt	both of the above
		[Ref. 16,17,18]

Along with the mine clearing equipment listed above the Soviet Union possesses many other items in the inventory to

deal with mines. Some of the items include, hand-held mine detactors, bangalore torpedoes, explosive line charges, and rocket propelled line charges. It is rather apparent that the Soviet Union takes the necessity for counter-obstacle capability very seriously. It is with this myriad of equipment, employed in effective combined arms operations that the Soviets plan to breach the NATO defenses in Western Europe.

C. EMPLOYMENT OF SOVIET FORCES

> "The offensive is the basic form of combat action. Only by a resolute offensive conducted at a high tempo and to a great depth is the total destruction of the enemy attained."
>
> General V.G. Reznichenko "Tactics" [Ref. 17: p. 3-1]

In the previous section, the vast quantities of Soviet counter-obstacle equipment were discussed. A discussion of the tactics that the Soviets are expected to utilize in order to accomplish their desired goals is contained herein. A key aspect in the employment of the Soviet forces is to capitalize on the high speeds of advance made possible with mechanized forces. The mission of the Combined Arms Army or Tank Army is to drive swiftly and deeply to the NATO Corps rear boundary. [Ref. 17] If there is one dictum of the Soviet Forces it is to maintain the speed of the advance.

Current military readings indicate that the Soviet commander desires to maintain a speed of advance of 50 to 80 kilometers per day under nuclear conditions and 40 to 50 kilometers per day under non-nuclear conditions. In order to be able to maintain these speeds, it will be necessary to reduce the effects of obstacles so that they do not hinder the advance of the Soviet forces. The Soviet commanders will be able to do this with their forces because of two primary reasons:

1. Large engineer units are located well forward with the combat arms.
2. Many simple engineer tasks will be handled by the ground gaining combat arms.

The Soviet forces will probably attack on high speed avenues of approach with one Motorized Rifle Regiment (MRR) in the lead followed by two additional regiments in the following echelon. The forces will attack with a high rate of advance and will usually pass pockets of resistance in order to maintain the high rate of advance.

The Soviet commander feels that in order to maintain a successful offensive operation, a numerical advantage of 3 to 1 is necessary. A Front will generally be utilized for the strategic offensive. This war-time only structure

usually consists of three or four Combined Arms Armies and one or two Tank Armies. The Front will attack with a structure similar to the following:

First Echelon	Three Combined Arms Armies (CAA)
Second Echelon	One CAA or One Tank Army
Front Reserve	One Tank or Motorized Rifle Division

The objectives of the Front are of three types, Immediate, Long Range, and Final. The definition of these are as follows:

Immediate	250-280 km	3-5 days
Long Range	500-560 km	7-13 days
Final	1000 km	15-21 days

With the strategic view as discussed above, the discussion will now switch to the smaller unit levels for a look at the tactics. It is important to keep in mind the Soviet desire to maintain the speed of the attack. In light of this, as mentioned earlier, normally each forward platoon will have two tanks fitted with mine plows. This allows the platoon to make two parallel breaches simultaneously. [Ref. 19] As the lead reconnaissance elements move towards the NATO defenses, the elements will find the minefields either by detonating mines, visually, or with vehicle mounted mine

detectors. Depending on the situation, the elements may mark the minefields or they may clear lanes with the organic mine-plows and rollers. Engineer personnel may affix hose charges to the mine-plow vehicles to pull through the minefields. When the hose charge is detonated, the area between the cleared tracks from the plow or roller is also cleared of mines.

If the reconnaissance elements do not locate the minefields, then the follow-on units may. They will probably locate them by detonating a mine. At this point the commander has basically three courses of action. They are:

1. Divert off the desired axis of advance around the minefield.
2. Conduct a hasty breach with organic plows.
3. "Bull through" with whatever assets are available.

When the Soviet commander has Engineer units with him, then the most probable option will be to "storm" the minefield with the engineers in the lead. This operation is a combined arms operation with tanks providing direct fire, artillery providing indirect fire and smoke, and the organic plows clearing lanes under engineer supervision. The standard method is to use explosives to augment the

breaching operation. [Ref. 13,14] Ideally, for a leading battalion on the main axis of the assault some 6-8 lanes may be ordered, one for each platoon [Ref. 17,18].

Other obstacles to Soviet movement, such as rivers or blown bridges, can be overcome by utilizing the the vast array of Soviet bridging equipment. Tank ditches or road craters can be countered by the bulldozer blade equipped tank.

This brief description of the Soviet counter-obstacle tactics and capabilities is as they exist today. The description is by no means complete, as that would fill several volumes. There exist many sources for information in this area. There is some problem though, in that many of the sources do not agree exactly with each other. The one area that seems to have universal agreement though, is that the capability of the Soviet forces is awesome. Perhaps this is due to the fact that the Soviet leaders recognize the necessity to continuously upgrade their forces and equipment.

The research for this chapter provides the basis for the concepts that are modelled for the Soviet forces in the STAR Engineer Effects Module.

The next chapter will discuss and describe some of the engineer effects modelling efforts that presently exist in the combat modelling community. These models have provided some useful ideas in the effort to provide an Engineer effects module for STAR.

IV. OTHER MODELLING IDEAS

This chapter will briefly discuss the relevant areas of three models currently in existence. The information available to the authors lacks computer code and is very general in nature. The purpose of this chapter is to highlight some existing modelling ideas that were of some use in the construction of the STAR Engineer Effects Module. The framework for this discussion is not to explain each model in detail, but to highlight each in the following areas:

1. Obstacle representation
2. Actions at obstacles
3. Actions out of obstacles

A. COUNTERCOM

The first model examined was COUNTERCOM, a combat simulation, developed by BDM Corporation, for evaluating the worth of military equipment and tactics in Company and Battalion level Armor/Mechanized Infantry warfare. [Ref. 20]

1. Obstacle Representation (COUNTERCOM)

COUNTERCOM models the minefield in a patterned representation. The minefield has several rows in a

minefield segment and each row is separated by an "inter-row spacing". Each row is a series of points (mines) separated by a fixed intermine spacing.

2. Action at Minefields (COUNTERCOM)

COUNTERCOM models mine rollers as a breaching device with an effectiveness parameter attached to count mine hits on the roller. The documentation pertaining to mine rollers does not explain the rollers representation or how its effects are modelled in the minefield. One other item of interest though, is how COUNTERCOM offsets following vehicles around dead tanks in the minefield. This is a desirable action to be modelled, however, the vehicles are allowed to pass too close to each other and thus, the tracks of the passing vehicles would actually touch the vehicle being passed.

3. Actions Out of Obstacles (COUNTERCOM)

COUNTERCOM models the SLUFAE (Surface Launched Unit Fuel-Air Explosive) mine clearing system as well as mine clearing line charges. These systems are Engineer unit employed and are beyond the scope of this dicussion. Future efforts to install the actual Engineer units and equipment in STAR may find this pertinent to that effort. The bypass

tactic is not modelled in COUNTERCOM. In this model, Engineer units are not modelled, however, certain Engineer equipment is. Indirect fire is played against the defender but not against the attacker. The deliberate breach does not suffer the effect of indirect fire.

B. MINEFIELD AND BARRIERS COMBAT SIMULATION

The second model examined was the Minefield and Barriers Combat Simulation, developed by a British firm, SCICON. [Ref. 21]

1. Obstacle Representation (SCICON)

SCICON uses a row minefield similar to that previously mentioned for COUNTERCOM. The start point for the mines is also Monte Carlo'ed in this model. SCICON represents a scatterable minefield by assuming a totally random representation. The distance to a mine detonation for a vehicle is determined by locating the mines in the vehicles path. The mean number of mines which fell within the vehicle's lane is found by using the density of the mines and the area of the path. This number may be used or the actual number of mines could be sampled from a binomial distribution. These mines are then placed randomly in the path of the vehicle. It is felt by the authors that the

average distance to a mine detonation could be determined from the reciprocal of the product of the minefield density and the vehicles width. This expected distance could serve as the mean of an exponential distribution draw. This would yield more realistic results and allow the model to benefit from the memoryless property of this distribution.

2. Actions at Obstacles (SCICON)

SCICON is reputed to play plows, rollers, and plow/roller combinations, however, from the information available it is difficult to ascertain their representation and their use within the minefields.

3. Actions Out of Obstacles (SCICON)

This model play simulates both the breach and the "bull through" options through obstacles. It is difficult to discuss how this is accomplished due to a lack of computer code. In the documentation, however, the divert tactic and how it is modelled is discussed at great length. A route to circumvent an obstacle is planned in advance of the simulation. This route is called a branch route stream. The branch route stream option is triggered at a barrier decision point. A barrier decision point is a point on the route, a route node, that is located, such that, it precedes

a known barrier. At this decision point, units stop to determine which actions to take in order to negotiate the obstacle. This is a kind of pseudo-dynamic movement, planned for on input, but implemented by tactical outcomes. This is a useful concept and its implementation in STAR would pave the way for dynamic route selection.

C. CASTFOREM

The final model to be discussed is CASTFOREM (Combined Arms and Support Task Force Evaluation Model), developed by the U.S. Army Tradoc Systems Analysis Activity. [Ref. 22] This is a stochastic, event sequenced, opposing force simulation of ground combat. This model has an Engineer process module that plays both organic and non-organic engineering capability and virtually any engineering task for which the user has the performance data. This model will be discussed in a different format than the previous models. This is due to the fact that this model is still under construction and the information available is in outline form. CASTFOREM is projected to be able to model Engineer tasks explicitly.

1. Methodology

* Task feasibility checked

  Personnel availibility

  Equipment availability

  Locational resources availability

* Task time simulation

* Task assessment is performed

  Resource integration

  Resource disintegration

2. Tasks Modelled

Emplace minefield

Emplace road crater

Emplace AVLB

Breach minefield/road crater

Construct protected positions

It appears on examination that this model will be quite data intensive in that it will demand detailed decision logic tables and performance data. It's general framework is complicated, yet it does not currently possess the capability to model tank ditches or bridge demolitions. No doubt this will be a capability in the future.

D. CONCLUSIONS

In order to avoid reinventing the wheel, the current state of the art in the community has been examined and briefly described herein. This preliminary research has provided several useful ideas and given the authors a perspective on the current state of the modelling of Engineer effects. The next chapter consists of the detailed description of the minefield obstacle model and its associated counter-obstacle tactical responses.

V. THE MINEFIELD MODEL

A. INTRODUCTION

This chapter discusses the simulation of the effects of a minefield in the STAR Combat model. Implicit in this task is the construction of the minefield and the modelling of its effects on the combat process. This model allows minefields to be emplaced, cause casualties, impede the attacker, and cause the attacker to execute predetermined tactical options.

B. MINEFIELDS MODELLED

In order to model the effects of a minefield on the tactical situation on a battlefield it is first necessary to construct the model of the minefield. However, prior to the discussion of how the minefields are modelled, a brief discussion of what is used to model the minefields, the STAR field module, is in order.

The framework for obstacle simulation in STAR is the field module. Fields are represented as elliptical areas on the battlefield which influence battle actions and outcomes. In order to cause actions for entities to take place, their

relation with respect to fields must be known. Rather than have actions take place at a specific point in time, actions are triggered by certain interactions of entities with fields. These interactions are results of either field boundary crossings, entries and exits, or field internal actions. The manner in which a boundary crossing or an internal action is activated takes place in the STAR MOVE routine. In the MOVE routine, each time an entity moves, the field boundary distance (FLD.BDY.DIST), to the nearest field is decremented. When this value reaches zero, then a field boundary action takes place. Similarly, when an entity is already in a field, a field internal distance (FLD.INT.DIST), the distance to the next field internal action, is also decremented. When the FLD.INT.DIST of an entity reaches zero, an internal action is initiated. Both these distances are attributes of moving entities on the STAR battlefield. Obstacle representation and the simulation of the resulting synergistic effects became a reality through the use of the existing STAR Field Module. For a more detailed description of this Module and its function in the STAR model the reader is referred to Reference 3.

The first section of this chapter contains information pertaining to the actual modelling of a minefield in STAR. In this model, five types of minefields are simulated, they include, the anti-vehicular minefield both patterned and scatterable, the anti-personnel minefield both fragmentation and blast, and the Claymore.

1. The Anti-Vehicular Patterned Minefield

The patterned anti-vehicular minefield consists of a series of belts with mines located on the belts. These belts may be considered as lines with the mines represented as points on the lines. In this respect, the mines are considered to be dimensionless. The belts are numbered sequentially as they are encountered. The odd numbered belts have mines located at even numbered mine locations, i.e. at 2,4,6,... meters. The even numbered belts have mines located at odd numbered locations, i.e. at 1,3,5,... meters. Thus, each belt has a constant density of .5 mines per meter of minefield frontage. The standard planning packages for U.S. patterned minefield densities are multiples of this constant density. [Ref. 7] In addition, the use of .5 also makes the simulation of the mine's location with respect to the vehicle much easier to

accomplish. Each belt simulates three rows of a minefield. In an actual minefield, three separate rows placed in a span of approximately 16 meters forms one belt. In order to simplify the process of modelling this activity, one belt of this model actually represents the projections of the three rows onto one single line.

This basic belt construct is used in conjunction with the existing STAR field module, to model the patterned minefield. [Ref. 3] The basic idea is that the mine belts will be superimposed on the field ellipse. The density of mines (per meter of front) of the minefield is used to determine how many belts are necessary. Once the number of belts is determined, then the size of the field ellipse needed can be determined. The size of the ellipse is computed in the following manner:

The semi-major axis equals the total width of the minefield divided by 2.

The semi-minor axis is determined by the formula: number of belts plus 1 multiplied by the belt spread in meters minus 10 meters.

This total is then divided by 2.

The resulting value is the length of the semi-minor axis of the ellipse. In symbols:

semi-minor axis = (((num belts +1)*(belt spread))-10)/2

The following sketch of an elliptical minefield with the mine belts superimposed should help to clarify the construction.

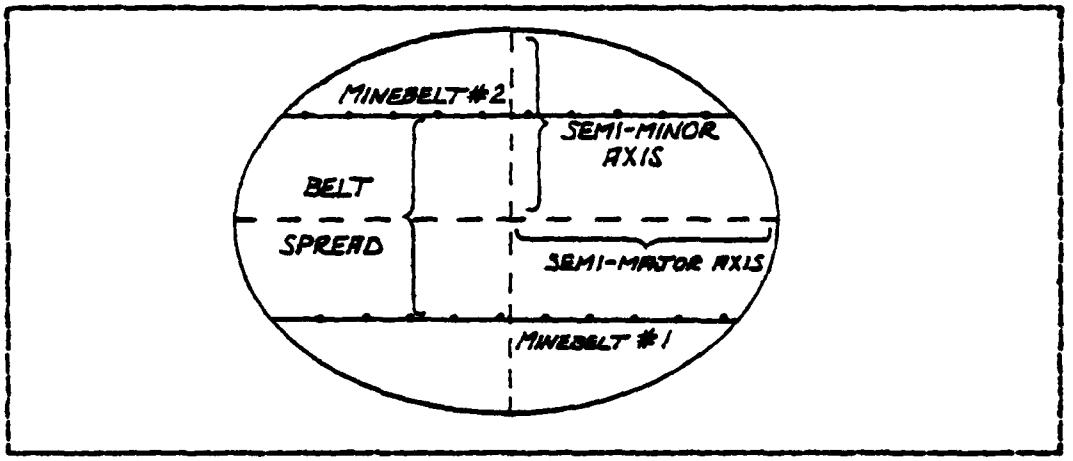


Figure 1: Minefield Representation

With the previous formula in mind, an example will be worked. A minefield with a density of 1.5 mines per meter of front, a belt spread of 50 meters, and 500 meters wide is desired. Using the formulas above the semi-major axis will first be computed.

1. The semi-major axis of the ellipse is the width of the minefield divided by two, thus, in this case it is 500/2 or 250 meters.

2. The formula for the computation of the semi-minor axis length yields the following:

   a. number of belts required equals the density desired divided by the density in each belt. For this minefield it is 1.5/.5 = 3
   b. ((number of belts+1)*belt spread-10)/2 = ((4*50)-10)/2 = 95 meters.

Thus, for this minefield the semi-minor axis is 95 meters and the semi-major axis is 250 meters.

Finally, in modelling the patterned minefield two assumptions are made:

The minefield is homogeneous in nature.

The mines may be played in the model in various states of masking, these states range from buried and visually undetectable to being placed on the ground and therefore, subject to avoidance.

2. <u>The Anti-vehicular Scatterable Minefield</u>

The anti-vehicular scatterable minefield is represented as a uniform distribution of mines throughout the area an elliptical field. The basic concept of the simulation of a scatterable minefield is that an expected distance to mine encounter is determined. The technique for determining the distance to a mine encounter will be described in detail in the following section. In order to emplace a minefield the following steps are involved:

a. Select the desired minefield size by specifying the semi-major and semi-minor axis lengths.

b. The minefield density is computed by dividing the number of mines in the minefield by the area of the ellipse.

In symbols: Minefield Density

$$= \frac{\text{number of mines}}{\text{area of ellipse}}$$

c. The expected distance to a mine encounter is equal to the reciprocal of the width of the vehicle times the density of the minefield.

In symbols:

Expected distance to encounter
= 1.0/vehicle width*minefield density

The expected distance to a mine encounter is then used as the mean value for a sampling from an exponential distribution. The value that is drawn is the distance to a mine encounter for the particular vehicle concerned. Repeated draws in this manner yield the distances to each successive mine encounter and the vehicle moves through the minefield in this fashion. This minefield is also assumed to be homogeneous in nature.

3. The Anti-personnel Blast Minefield

The anti-personnel blast minefield is represented in a manner that is similar to the methodology utilized for the Anti-vehicular scatterable minefield. The mines are assumed to be uniformly distributed throughout the area of the elliptical field. The minefield density is computed in the same manner as the scatterable minefield discussed above. The expected distance to a mine encounter for a dismounted infantry entity is then determined by utilizing the Army Small Arms Requirements Study (ASARS) formula. A random number draw from a Uniform (0,1) distribution is made and the draw is then substituted in the following formula:

Distance to a mine encounter
= -(2.5)*ln(1-Draw)/minefield density*foot width [Ref. 23]

If the entity is a vehicle, then the distance to a mine encounter is computed by drawing from an exponential distribution with a mean equal to the expected distance for a vehicle mine encounter. This expected distance is equal to three divided by the product of the minefield density and the width of the vehicle. An assumption is made that the tracks/wheels of a vehicle constitute one third of the vehicle width.

Expected distance for vehicle mine encounter
= 3.0/minefield density*width of the vehicle

Thus, the entity moves through the field by distances from mine encounter to mine encounter.

4. <u>The Anti-personnel Fragmentation Minefield</u>

There are two types of anti-personnel minefields considered in this model. The types are the regular fragmentation mine and the Claymore.

a. Regular Fragmentation Mines

The regular anti-personnel fragmentation mine is represented as a circle with a square superimposed on it. The tripwires of the mine are represented as extensions of the diagonals of the superimposed square. (See Figure 2) The mines are placed in an orientation that favors the attacker. This is accomplished by orienting the mine so

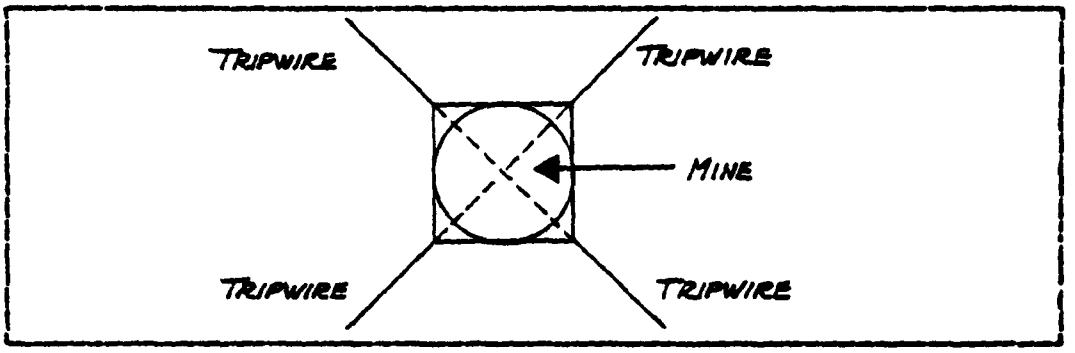


Figure 2: Representation of fragmentation mine

that the attacker will always hit the tripwire at a 45 degree angle. The actual length of the tripwire exposed to the attacker is the product of the cosine of 45 degrees and the length of the tripwire. The reason for this conservative action is to allow for cases in real life when the tripwire does not deploy as is intended by design. Thus, the total tripwire length that is exposed to the attacker is really the projections of the effective lengths of each tripwire onto a line perpindicular to the direction of movement of the attacker. The length of the tripwires available to detonate the mine (effective triggering width) is equal to two times the effective length of the tripwire. (See Figure 3) The minefield density is found in exactly the same manner as for the scatterable minefield.

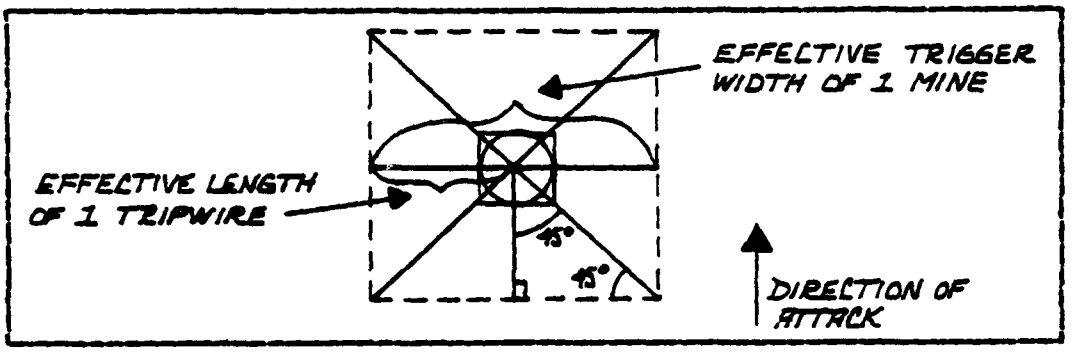


Figure 3: Effective Tripwire Length

The expected distance to a mine encounter is computed by using the reciprocal of the product of the minefield density and the effective trigger width of one mine. This expected distance is then used as the mean for a sampling from an exponential distribution in order to determine the actual distance to the mine encounter. Vehicles are assumed tc not be subject to the effects of a fragmentation mine, and thus, need not have expected distances computed; They are allowed to move freely through the minefield with no effects from the mines. Vehicles do, however set the mines off and thus, decrease the density of the minefield. In this iteration of the module, dismounted infantryman in the area of anti-personnel fragmentation mines detonated by vehicles, are not subject to assessment as casualties from the detonations. The dismounted infantrymen are however, subject to the

effects of mines detonated by dismounted elements in their respective platoons.

b. Claymore Fragmentation Mines

The effective casualty area of a Claymore mine is a 60 degree arc with a radius of approximately 100 meters. The activation area of a Claymore is represented by an ellipse with a semi-minor axis of 14 meters and a semi-major axis of 24 meters.

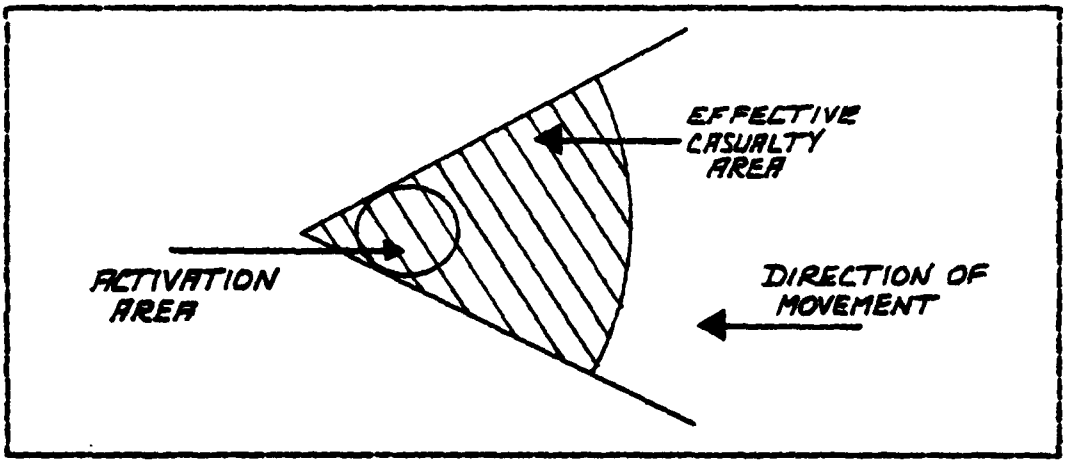


Figure 4: Claymore Mine Representation

The semi-major axis is oriented parallel to the direction the attacker is moving. Once the ellipse is entered, the distance to the mine encounter is determined to be twenty percent of the length of the semi-major axis. (See Figure 4)

When the entity has travelled that distance, the Claymore detonates and casualties are assessed based on the effective casualty area for a Claymore mine. Any member of the triggering entity's platoon within the effective casualty area of the Claymore at the time of detonation may be assessed as a casualty from the blast.

This concludes the general discussion of the representation of minefields in STAR. The discussion now centers on the subject of what the physical representation yields in the STAR model.

C. THE MINEFIELD IN STAR

The model of a minefield must produce two pieces of information, the distance an entity travels prior to a mine encounter and the consequences of that mine encounter. In order to understand how this information is obtained, the following discussion explains the function of the minefield in STAR. In the MOVE routine, the model is continuously computing the distances to fields, so that appropriate actions may take place. These actions may take any of three forms, field entries, field internal actions, and field exits. During the discussion of the three types of actions, several key routines are also described. A thorough

understanding of these key routines is a necessity if one is to understand this simulation of Engineer effects on the combat process.

1. Minefield Entry Actions

In the model, as an entity crosses a field boundary a test is made to determine the type of field being entered. Once the field is determined to be a minefield, then the distance to a mine encounter is computed. This computation takes place in the routine named MINE.SCHEDULE. (See Figure 5)

a. Routine MINE.SCHEDULE

In MINE.SCHEDULE the distance to a mine encounter will be determined. First, the the major category of the minefield is determined.

(1) Anti-vehicular Minefields. The second action is a determination of the system type of the entity that crossed the boundary. If the entity is a dismounted infantryman, then the minefield has no effect on the entity and he is allowed to proceed through the minefield. This is due to the assumption that a dismounted infantryman is not able to activate anti-vehicular mines. If the entity is a vehicle, then the mine type is checked. The mine type

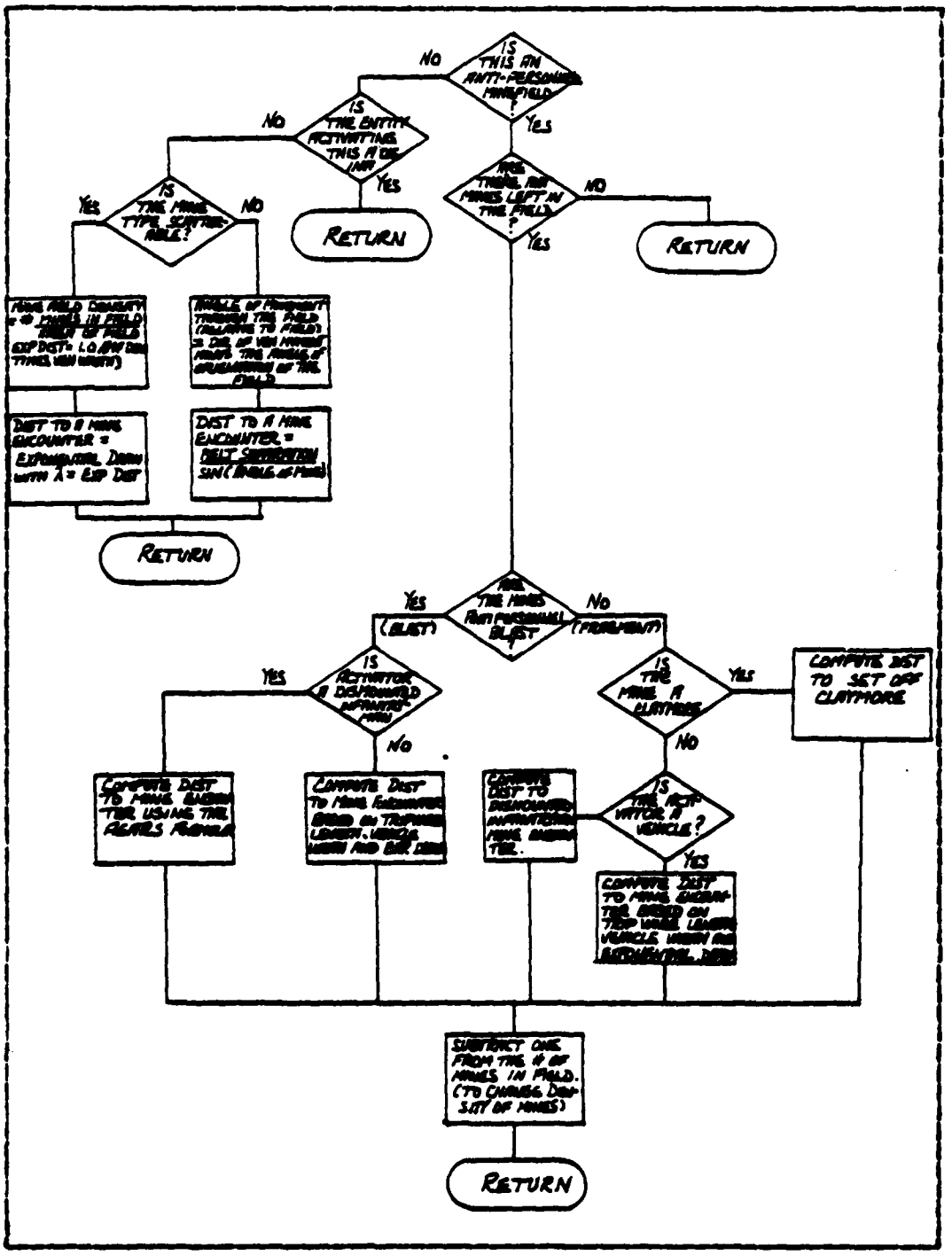


Figure 5: Routine MINE.SCHEDULE Flowchart

dictates whether the minefield is scatterable or patterned. Once the type minefield is determined, then the distance to a mine encounter is computed.

If the type of minefield is scatterable, the computation of the distance to a mine encounter takes place as follows:

1. The minefield density is determined by dividing the total number of mines in the field by the area of the field.

2. The width of the appropriate vehicle is obtained from the vehicle dimension tables.

3. The expected distance to a mine encounter is equal to 1.0/(width of vehicle*minefield density).

4. The actual distance to the mine encounter is determined by sampling from an exponential distribution with mean equal to the expected distance computed above.

If the minefield is a patterned or hand-emplaced minefield the distance to the mine encounter is determined as follows:

1. The direction of movement is determined for sign i.e. is it positive or negative. If the direction is negative, then it is converted to the appropriate angle on a 360 degree reference system. For example, -90 degrees is equal to 270 degrees.

2. Once the direction of movement is converted, it is then translated to the coordinate system relative to the orientation of the minefield. The angle necessary for the translation is called THETA, and will be used in further computations

3. The distance to the minebelt encounter is equal to the belt spread divided by the absolute value of the sine of THETA.

(2) Anti-personnel Minefield. If the minefield type is determined to be anti-personnel, then the following actions take place:

a. The minefield is checked to see if there are still any undetonated mines remaining in the field. If all the mines have been detonated, then the routine returns with no further action.

b. If the minefield is still an active casualty producing field, then a test is made to determine if the mine type is blast or fragmentation

1. If the field is a blast type minefield, then the activator is checked to determine is a dismounted infantryman. A dismounted infantryman receives a distance to a mine encounter computed by using the ASARS formula. If the activator is a vehicle, then the distance to a mine encounter is computed using the vehicle dimensions, minfield density, and a sample from an exponential distribution as previously described.

2. If the mine is a fragmentation mine, then a check is made for Claymore mines. If the mine is a Claymore, then the distance to activation is

computed. When the fragmentation mine is not a Claymore, then the activator is identified. The expected distance to a mine encounter for a vehicle is based on the reciprocal of twice the effective tripwire length plus the vehicle's width. This expected distance is used as the mean of an exponential distribution sampling. For a dismounted infantryman, the distance to a mine encounter is computed based on the tripwire length and a sampling from an exponential as discussed previously.

c. Finally, the density of the minefield is decreased for the next activator by subtracting one from the total remaining mines in the field.

Thus, the action at a minefield entry will return the distance to the next mine encounter for the scatterable anti-vehicular minefield, the distance to the next mine belt for the patterned anti-vehicular minefield, and the distance to a mine encounter for both anti-personnel minefields and the distance until detonation/activation for a Claymore mine. The entity attribute FLD.INT.DIST is now assigned the value of the distance returned from

MINE.SCHEDULE. Once the entity travels this distance, then the next action, the field internal action takes place.

2. Minefield Internal Actions

The entity has travelled the appropriate FLD.INT.DIST in the MOVE routine. An internal action for a mine encounter now occurrs. During a minefield internal action, several actions may take place. In general, the actions include such events as the slowing of the speed of the vehicle, the lowering of the mine plow, the assessment of mine hits on the vehicle, and the altering of the platoon's formation. In this section of the chapter, the assessment of mine hits on a vehicle is discussed. The remaining internal actions are discussed in the detailed description of the minefield internal actions in section E.2.b. of this chapter. During an internal action Routine POP.A.MINE is called. (See Figure 6)

a. Routine POP.A.MINE

POP.A.MINE does the following:

1. Determines the number of mines hit, if any.
2. Determines the location on the vehicle of any hits, i.e. a belly or a track hit and the distance from the mine detonation if the entity is a dismounted infantryman.
3. Determines the damage from the mine encounter based on the mine lethality data.

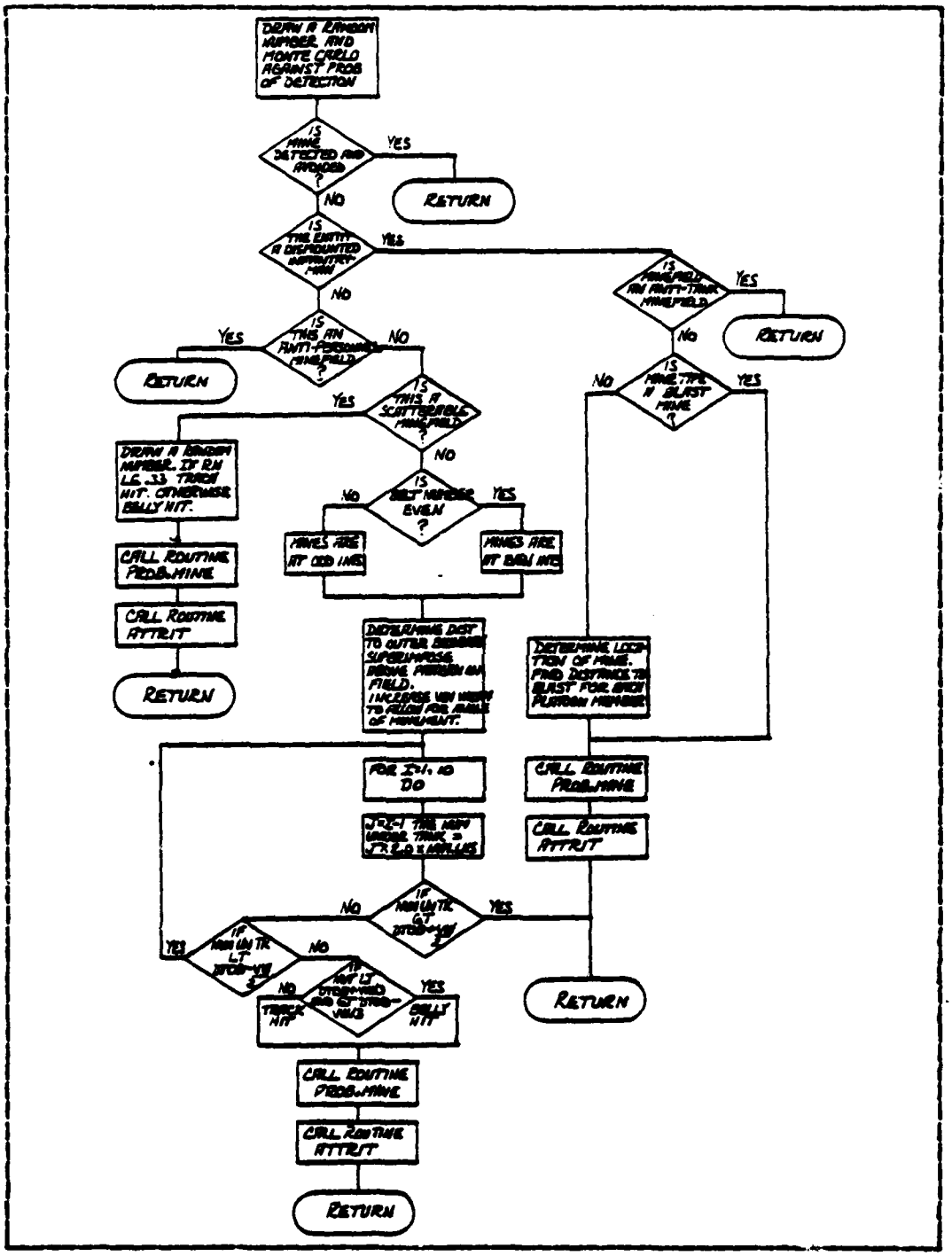


Figure 6: Routine POP.A.MINE Flowchart

In routine POP.A.MINE, a test is made to determine if the mine can be detected and avoided, (a scatterable or a hand-emplaced type that is not buried). [Ref. 24] If the mine can be detected, then a random number is drawn to determine if the mine is avoided. When a mine is avoided, then a new mine encounter distance is determined and the entity continues to move. The next internal action will occur at the new mine encounter distance. If, however, the mine is hit, then a check is made to determine if the entity is a vehicle or dismounted infantry.

If the field is an anti-personnel minefield and the entity is a dismounted infantryman, then the minefield is checked to see if it composed of blast or fragmentation mines.

1. If the minefield contains blast mines, then the soldier will die because he has just stepped on a mine. Routine ATTRIT is called to update the simulation with respect to the fatal disposition of the soldier that caused this action.

2. If the minefield contains fragmentation mines, then the exact location of the mine is determined. Then, the distance from the mine detonation to each member of the

activator's platoon is computed. If an entity is within the casualty radius of the mine, then routine PROB.MINE is called. Routine PROB.MINE accesses the mine lethality data for the mine detonation against personnel. This data is based on the type mine, the entity's posture at the time of detonation, (standing or prone), and the distance from the entity to the mine detonation location. The lethality data returns a probability of kill that is then Monte Carlo'ed to determine the soldier's casualty status. The result of the Monte Carlo is then sent to routine ATTRIT to update the simulation with respect to the soldier's status.

When the activating entity is a vehicle, then a check is made to determine the type of minefield entered. If the minefield is an anti-personnel minefield, then no damage is assessed against the vehicle. When the type of minefield is anti-vehicular, then a test is made to determine if the minefield is patterned or scatterable. If it is scatterable, then a random number is drawn from a Uniform (0,1) distribution to determine the mine detonation location on the vehicle. An assumption is made that one-third of the vehicle bottom area is track and two-thirds is belly. The random number is then Monte Carlo'ed against

this standard to determine the location of the mine hit. Once the hit location is determined, routine PROB.MINE is called. PROB.MINE accesses the vehicle lethality data for mine hits based on the mine type, vehicle type, and the hit location on the vehicle. PROB.MINE passes the appropriate lethality data to routine ATTRIT, the damage to the vehicle is assessed, and the simulation is updated.

If the mine type/minefield is patterned anti-vehicular, the following computations are made:

1. The number of the next belt to be hit is determined.
2. The distance to the outer boundary is computed.

The outer boundary is the tangent line on the ellipse that is perpendicular to the semi-major axis. This distance is computed in several steps. The first step is to convert the X coordinate of the location of the vehicle to an X prime location in a coordinate system that uses the major axis of the field as the X axis and the minor axis of the field as the Y axis. X prime is obtained by the translation formula:

X' = X*cos(rotation angle)+Y*sin(rotation angle)

The distance to the outer boundary (DTOB) is now calculated as the absolute value of the difference between the X coordinate value of the center of the field and the X prime

value described above. This number is then subtracted from the length of the semi-major axis of the field. This formula is applicable to both sides of the minor axis of the field. It is this distance that is the key to superimposing the mine belts on the elliptical field, and thus, the locations of the mines relative to the vehicle's position. The vehicle width (VW) is also adjusted to account for the angle of vehicular movement. Next, the integer mine location number just to the left of the vehicle (NUM.LEFT.SIDE) is determined. The routine then determines if the belt is even or odd, and consequently, if the mines are located at even or odd integer locations. The NUM.LEFT.SIDE is then adjusted to the appropriate odd or even integer further to the left of the vehicle. A loop search is then conducted to determine where the mines are located in relation to the vehicle; track, belly, or miss. Again, the assumption is made that the track takes up one third of the vehicle width. The detonation locations are found and routines PROB.MINE and ATTRIT are called to assess the damage. The routine PROB.MINE will now be described. (See Figure 7)

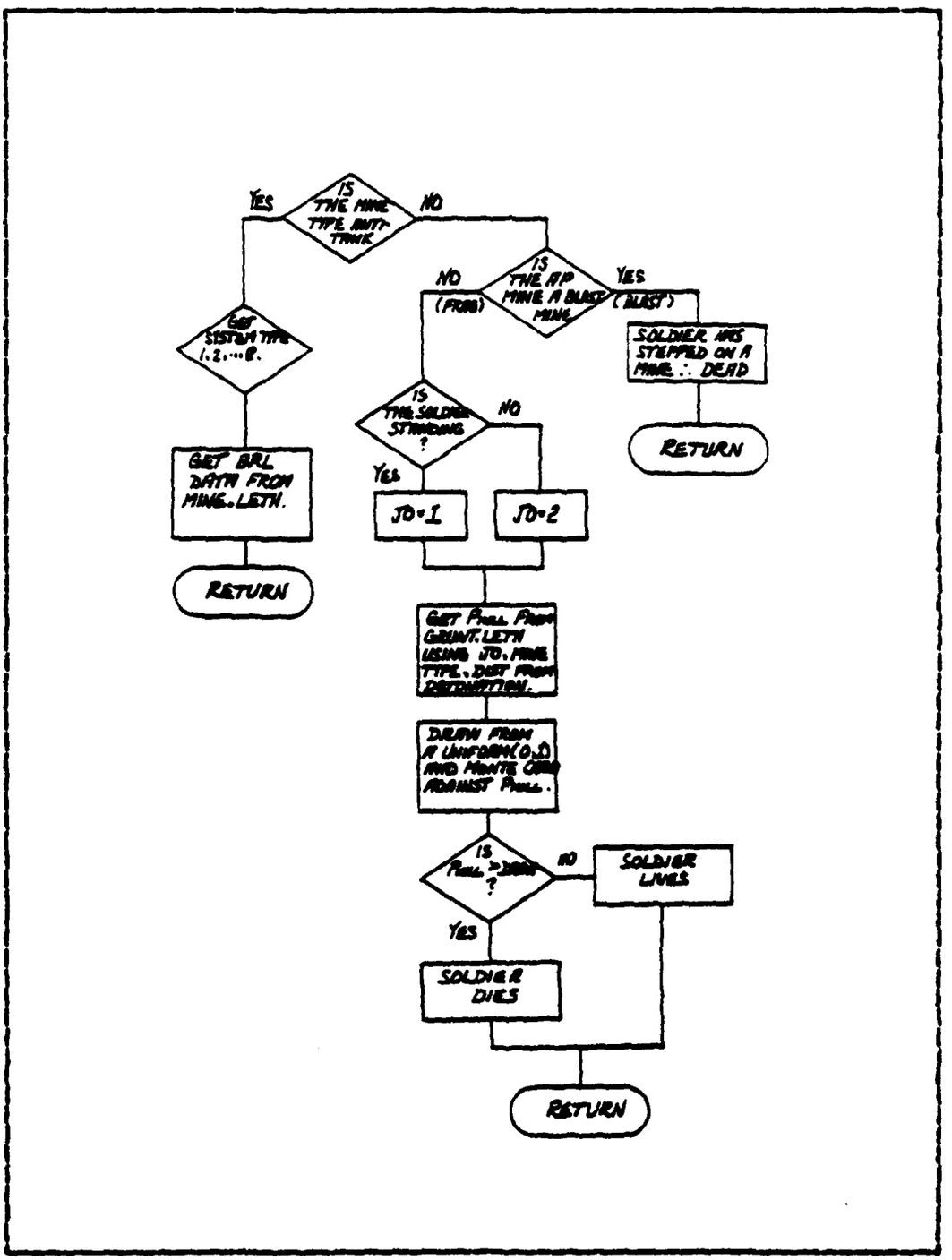


Figure 7: Routine PROB.MINE Flowchart

b. Routine PROB.MINE

The purpose of this routine is to access the appropriate mine lethality data and assess the damage to the entity.

For dismounted infantry the lethality data is stored in a three dimensional array called GRUNTLETH. The

Table 6: GRUNTLETH

Mine type	Posture of soldier	Distance from mine detonation
1 M16 2 ADAM 3 M74 4 Claymore	1 soldier standing 2 soldier prone	distance is in meters, except Claymore data is by 10 meter increments

dimensions of GRUNTLETH are four by two by five. The value that is returned from GRUNTLETH is a probability of kill.

If the mine is anti-personnel, then a check is made to determine the actual type of mines in the field. When the mine is a blast mine, then the soldier becomes a casualty. If a soldier activates a fragmentation mine then any member of his platoon may be affected. The posture of each soldier in the platoon, his distance from the mine detonation, and the type mine are used to access GRUNTLETH in order to determine a probability of kill. For each mine

detonation, a probability of kill for each live platoon member is determined. This probability is Monte Carlo'ed by drawing from a Uniform (0,1) distribution to determine if any soldiers are casualties as a result of the blast.

The mine lethality data for vehicles is stored in the array MINELETH. MINELETH is a four dimensional array that is five by eight by two by four. The data stored in MINELETH is Ballistics Research Laboratory (BRL) lethality data. It is stored as follows:

Table 7: MINELETH

Mine Type	Vehicle Type	Detonation Location	Damage Probabilities
1 M70	1 Tank	1 Track	Mobility
2 M56	2 APC	or	Firepower
3 M21	3 Command	2 Belly	Mob/Fire
4 M15	4 BRDM	Hit	Catastrophic
5 M19	5 Zsu		
	6 SP How		
	7 SA11		
	8 SPAT 125		

Routine PROB.MINE first tests to determine the mine type. If the mine is anti-tank, then the system type of the activating entity is determined. With the type anti-tank mine, vehicle type, and the detonation location on the vehicle, MINELETH is accessed and the appropriate damage probability values are returned to routine POP.A.MINE.

At this point, the program would return from routine POP.A.MINE and call routine MINE.SCHEDULE. A new distance to a mine encounter is determined and the cycle continues until the entity becomes a casualty or exits the minefield.

3. Field Exit Actions

When an entity exits a minefield its speed is upgraded to the maximum allowed by the terrain. If a lane has been cleared through the minefield, then this information is stored for future use by follow-on elements.

The previous discussion has described how the minefield is represented in the STAR model. In the next section, the discussion will focus on the synergistic effects modelled.

D. SYNERGISTIC EFFECTS

In this section, the tactics and synergistic effects of minefields are discussed.

1. Tactical Alternatives for the Commander

A tactical commander encountering a minefield has basically three alternatives to choose from:

a. He may choose to conduct a hasty breach if the unit possesses any live plows or rollers.

b. He may bypass the minefield if the tactical situation permits.

c. He may, as a last resort, order a "bull through" operation, just driving the vehicles through the minefield hoping to clear a lane and taking casualties.

The choice that the commander makes is based on whether or not he possesses knowledge of the minefield. The manner in which the knowledge is gained is irrelevant. What is relevant though, is when the knowledge is gained. If the commander gains the knowledge prior to entering the minefield, then there is more flexibility in the selection of tactics that are most beneficial for the mission of the unit. If, on the other hand, he gains his knowledge the hard way, i.e. by detonating a mine, then the commander is forced to react in a different fashion. In the design of this model, some judgmental decisions based on knowledge and experience have been made. As a result, certain tactical alternatives were chosen for simulation in this Engineer Effects Module. A recapitulation of the tactical alternatives in this module are included in Table 8.
The expected tactics of both the US and the Soviet force commanders are the basis for this model.

2. Minefield Synergistic Actions

The general minefield effects simulated will now be described. As mentioned above, the knowledge of the

Table 8: Tactical Alternatives at an Obstacle Encounter

1. Conduct a breaching operation through the obstacle whenever the platoon possesses the proper breaching equipment.
2. "Bull-through" the obstacle if specified by the user on input.
3. If the "bull through" is not specified, then the entity will:
   a. Move to a cleared lane through the obstacle, if one exists.
   b. Bypass the obstacle if a bypass exists and no clear lane exists through the obstacle.

minefield and the mission of the force are the driving factors in the selection of the counter-obstacle tactics. In this section of the chapter, the breach and "bull through" actions are discussed. The bypass action will be discussed in Section 3. of this chapter. It is for this reason that the discussion will begin with the knowledge attribute of the field. The knowledge attribute of the field may have any of four values, 0,1,2, or 3.

a. 0 or 1 means that there is no prior knowledge of the existence or location of the minefield.

b. 2 means that the minefield's location, but not its extent, is known and that partial lanes through the minefield may exist.

c. 3 means that the minefield, and any clear lanes through it are known.

(See Figure 8) As an entity crosses the field boundary into a minefield, the knowledge level on the field is checked and the appropriate action is taken.

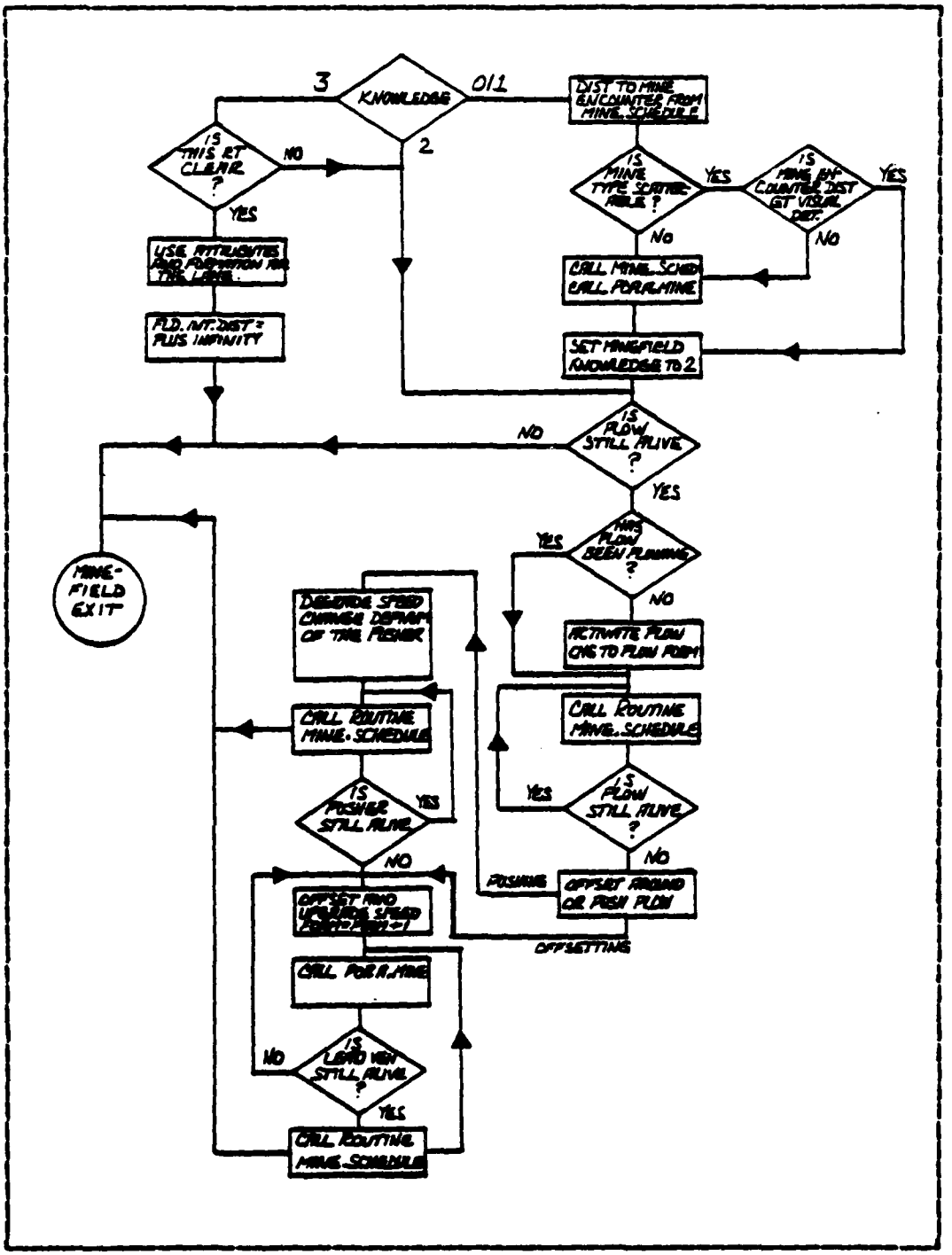


Figure 8: General Minefield Boundary Actions Flowchart

a. Actions with No Prior Knowledge

If the knowledge level is 0 or 1, then routine MINE.SCHEDULE is called to compute a distance to a mine encounter. If the minefield type is patterned or hand-emplaced, then once this distance is traversed by the entity, a detonation occurrs and the knowledge level is changed to two. If, however, the mines are visually detectable, then the possibility of avoiding the mine exists. If the distance to the mine encounter is greater than the minefield visual detection distance, then the minefield is allowed to be detected without a detonation occurring. The minefield visual detection distance is designated by the user. Two studies on the subject, the TEMAWS study and a study conducted at Ft Drum N.Y. [Ref. 25,26], may be helpful in the determination of this value. The distance for the entity to travel in the minefield is set to this visual detection distance, and the knowledge level on the minefield is set to two. If on the other hand, the encounter distance is less than or equal to the visual detection distance, then routine POP.A.MINE is called, the detonation occurs, and the knowledge level on the minefield is set to two.

The knowledge level is two, and if the entity is a vehicle the next check is to determine if there are any alive plows remaining in the platoon. If no plows are alive, then the platoon turns around and leaves the minefield on the same path it entered. The next action for this exiting platoon is a decision ellipse entry action. This action is discussed in Section E.1. of this chapter. If the platoon possesses a plow that is alive, then the platoon stops for a plow activation time and changes to a plow formation (the plow leading with all other vehicles following in column directly behind the plow). The platoon also has its speed degraded by a plowing speed factor.

In using a mine plow to clear a lane through a minefield several assumptions are made about plows.

1. Plows clear all mines from the path of the tracks of the vehicle.
2. The plow suffers no degradation due to mines hitting the skids.
3. Following tanks are able to follow directly in the tracks cleared by the plow as a result of the berms created.

Once plowing begins, routine MINE.SCHEDULE is called to establish a new mine encounter distance. If the plow vehicle is still alive when this point is reached, then routine MINE.SCHEDULE is called again to compute a new mine

encounter distance. This process continues until the minefield boundary is crossed or the plow vehicle is killed by direct fire. If the plow is killed during the travel to the next mine encounter, then there are two options for the platoon, the dead plow may be pushed by the following tank or the live vehicles may offset around the dead plow and attempt to make a lane through the field without it. When the plow vehicle is dead, then a draw from a Uniform(0,1) distribution is made. The random number is Monte Carlo'ed against a user specified probability of pushing, to determine which option, pushing or offsetting, is selected.

If the offset option is selected, then routine POP.A.MINE is called to assess mine damage to the lead tank at the location of the mine detonation. The speed of the platoon is upgraded to the maximum allowed by the terrain and routine MINE.SCHEDULE is called to determine the distance to the next mine encounter. At the mine encounter, routine POP.A.MINE is called and the damage is assessed to the lead tank. This process continues until either the platoon is depleted or the minefield is exited. If the push option is selected, the platoon's speed is degraded further from the current plowing speed and the platoon continues to

move through the minefield just as though the plow were alive. Several assumptions are made with respect to the pushing option.

1. Another tank can push a plow tank provided that the tracks are still on it. A plow tank killed by direct fire is assumed to still have its tracks mounted.

2. A pushing tank gains a slight advantage against direct fire weapons due to the fact that it is shielded somewhat, by the tank it is pushing.

3. A platoon can only elect to push the plow once. If the pushing vehicle is killed the option to push it and the plow does not exist.

4. The pushing tank is able to follow directly in the cleared track path of the plow.

Once the pushing has begun, routine MINE.SCHEDULE is called to compute the distance to the next mine encounter. At the encounter, the push tank is checked to see if it is alive. If it is still alive, then the process continues until either the platoon exits from the field or the pusher dies. If the pusher is dead, then the offset sequence begins and continues as previously described.

The next path to be described is the case when the knowledge level is two. Again, the reference will be to Figure 8.

b. Actions with Knowledge Prior to Entry

When the knowledge level is two, upon entry into a minefield, an entity need not detonate a mine to find out that a minefield exists. The platoon is already aware that a minefield exists and first checks for an alive plow in the platoon. After this, the path is the same as described in the previous section from the point where the knowledge level was changed to two.

c. Actions When a Lane is Known to Exist

The final case to be discussed for the minefield actions is the case when the knowledge level is three. This means that there is a clear lane somewhere through the minefield. This is the final case to be described from Figure 8.

A check is made to see if the route the platoon is travelling is clear of mines. If this is not the case, then the same path as a knowledge level of two is followed. The unit will attempt to gain a breach. The reason for this is that the commander takes actions to secure additional lanes through the minefield. The desire of threat unit commanders to secure at least one lane per platoon is discussed in Chapter 3 of this thesis. If, however, the

platoon is travelling on a cleared route, then the platoon is assembled into a column formation and the speed is degraded for lane travel safety. The formation and the lane speed factor are both input parameters. In this case, two additional assumptions are made:

The follow-on platoons will be able to follow directly in the cleared, marked lanes.

If a tank is killed by direct fire while travelling on the cleared lane, the following vehicles will be able to push the dead tank out of the way and continue on the lane.

Each of the above knowledge cases eventually yields the same result, a minefield exit. Basically, there are two actions that occur at a minefield exit. If the vehicle is exiting the minefield ellipse as the result of clearing a lane, then the knowledge level of the minefield is changed to three and the platoon conditions are reset for future minefield encounters. The knowledge attribute allows follow-on units to know that this particular route through the minefield is clear. If the vehicle is not exiting as the result of clearing a lane, then it is moving in a reverse direction back along its route towards a field ellipse called a decision ellipse. The decision ellipse and its function in the module is discussed in detail in the next section of this chapter, Section D. 3. In this case,

the speed and formation are upgraded to the original formation and speed but a cleared lane is not recorded.

3. The Decision Ellipse

In the previous section, reference was made to a new type of field, the decision ellipse. This creation is the key to the capability to insert pseudo-dynamic movement into the STAR combat modelling process. The decision ellipse allows the elements in the simulation to make route changes based upon the tactical situation and the status of the platoon's organic breaching equipment. The movement has been labeled as pseudo-dynamic because the alternative route options must be planned in advance as input, however the actual selection from the alternatives is dictated by the battlefield situation.

The decision ellipse is located on a route such that it will be encountered prior to an engineer obstacle. The decision ellipse is to be located such that it affords the attacker maximum cover and concealment from enemy direct fire. At the decision ellipse, the first decision concerning the minefield is made.

The general actions that take place at a decision ellipse will now be discussed. (See Figure 9)

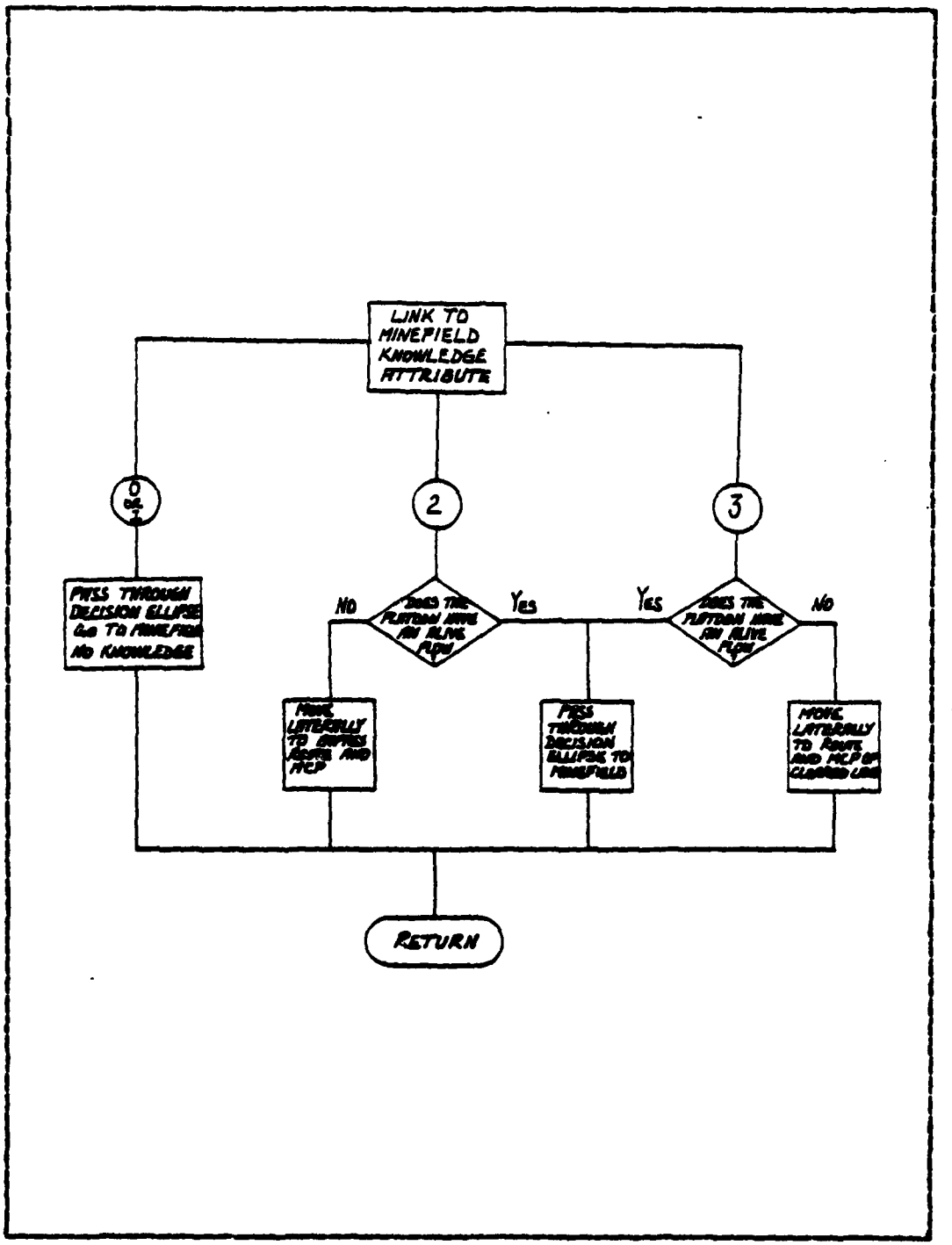


Figure 9: General Decision Ellipse Flowchart

The decision ellipse is linked to the obstacle ellipse that is associated with it. This is accomplished by the use of an integer obstacle ellipse name. As a vehicle moves into the decision ellipse, the knowledge level of the obstacle is ascertained by linking to the appropriate obstacle ellipse. If the obstacle ellipse knowledge level is 0 or 1, then the vehicle passes through the decision ellipse with no further action. If the knowledge level is two, then the platoon is checked for the presence of a plow. If the plow is alive, then the platoon continues to move with no further action in the decision ellipse. If the plow is not alive, then the platoon is given its bypass route and next movement control point (MCP) and moves toward the new route. If the knowledge level is three, the platoon is checked for the presence of an alive plow. If the plow is alive, the platoon will continue to move on its route through the decision ellipse with no further action until the minefield boundary is crossed. If the plow is not alive, then the platoon will obtain the route and MCP of the cleared lane and move to it.

In order to better illustrate the concept of the decision ellipse the reader is referred to Figure 10 for the following discussion of some situations.

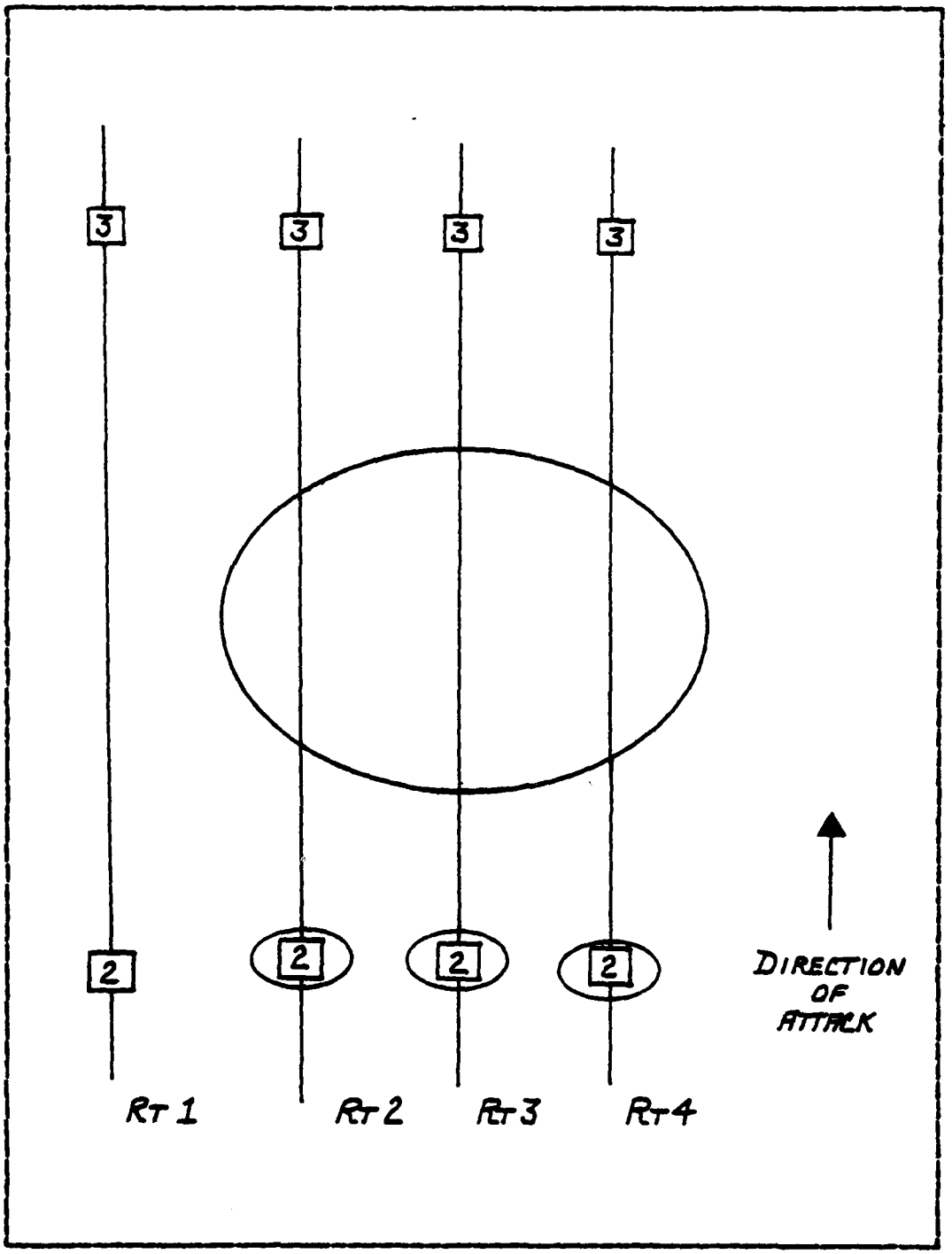


Figure 10: Decision Ellipse Example

Each route has a platoon travelling on it. The decision ellipses were set up in the following manner:

Decision Ellipse	Bypass Route	Bypass MCP
A	1	2
B	2	2
C	3	2

Situation 1:

RT 3 is a cleared route through the minefield. The platoon travelling on RT 4 has reached the decision ellipse and does not have a plow vehicle alive. The knowledge level of the obstacle is 3, thus, there is a cleared lane through the minefield. The decision ellipse tells the platoon to go to the cleared lane located at RT 3 and MCP 2.

Situation 2:

The platoon on RT 3 reaches decision ellipse B and their plow is dead. The knowledge level on the obstacle is 2 because other platoons have entered the minefield and are attempting to clear lanes through it. The platoon on RT 3 does not have a clear lane or a plow, therefore it will go to the bypass, RT 2 and MCP 2. As the platoon reaches decision ellipse A on its bypass route it will again check to see if a lane has been completed through the minefield.

If there still is no lane through the minefield, the platoon will bypass to RT 1 and MCP 2. These bypass routes and MCPs are kept as attributes of the entered decision ellipse. If however, the knowledge level is 3, because a lane had been completed while the platoon was moving from B to A, then the platoon will move to the route that contains the cleared lane. The location of the information about the cleared lanes is carried in a three-dimensional array called TRAFFIC.CONTROL. The following information is stored for each route for each obstacle:

1. Route number of the route being travelled.
2. Decision ellipse MCP on that route.
3. Cumulative distance clear in the obstacle on the route.
4. The most recent distance to a mine encounter for the lead vehicle of the lead platoon in the minefield.
5. The number of mine encounters scheduled thus far for the elements travelling on this route through the minefield.

For a more detailed discussion of the TRAFFIC.CONTROL array, see Appendix A.

In this section, the discussion has been of a general nature with regard to what is modelled. This is a good foundation and a necessary prerequisite for an understanding of the next section, which consists of a

detailed description of the actions that take place at the various locations pertaining to fields.

E. DETAILED FLOWCHART APPROACH

In this section the discussion is divided into three categories, actions at field entry, field internal actions and actions at field exits. Section C of this chapter, gives the reader an overview of the synergistic effects captured in this model. This section discusses in further detail how the desired synergistic events are captured in this model of the combat process. The reader is advised that routines POP.A.MINE and MINE.SCHEDULE play an important role in this discussion and thus, should be understood well, prior to continuing. This model is constructed in a fashion to account for many simultaneous actions occurring on the battlefield. For example, several platoons may be moving through the same minefield either on different or common routes or combinations thereof, concurrently. The ability to handle many simultaneous actions necessitates the use of many flags and tests in order for the model to discern which entities are involved in the actions. It is for this reason that the reader will notice a great number of tests and different situations covered in the following discussions. The decision ellipse will be discussed first. (See Figure 11)

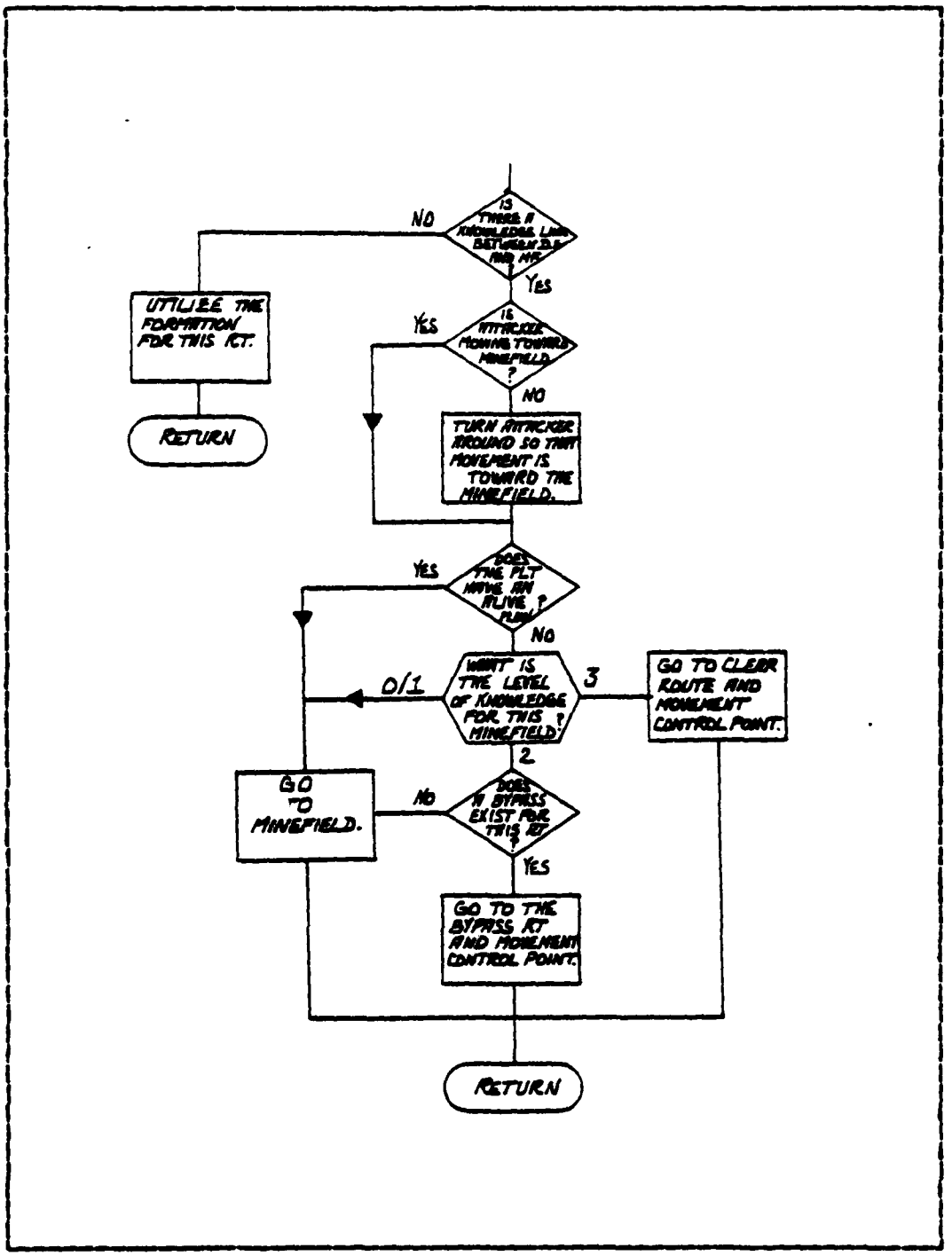


Figure 11: Minefield Decision Ellipse Flowchart

1. The Decision Ellipse (Detailed)

The decision ellipse on each route carries the following additional attributes as information:

Table 9: Decision Ellipse Attributes

P1.FLD	The bypass route for this route.
P2.FLD	The bypass MCP on the bypass route.
P3.FLD	The field number of the obstacle associated with this decision ellipse.
P4.FLD	The number of the route this decision ellipse is located on.
P5.FLD	The MCP inside the decision ellipse.

When an entity enters a decision ellipse, the decision ellipse attempts to link to the minefield ellipse in order to obtain the knowledge level of the minefield. The link is possible because the decision ellipse is given the integer number name of the minefield it is associated with as an attribute. If the decision ellipse is able to link to the minefield, then the route through the decision ellipse also goes through the minefield. In order to ensure that the next action is a decision ellipse exit action, the internal distance for the vehicle is set to plus infinity. If the link is not possible, then the decision ellipse is on a route that is a bypass around the minefield. If the entity is on a bypass route, then the entity must utilize the formation on the route. However, if the entity is

utilizing lateral movement in order to circumvent the obstacle, then the route formations are not used. This instance requires a more detailed examination.

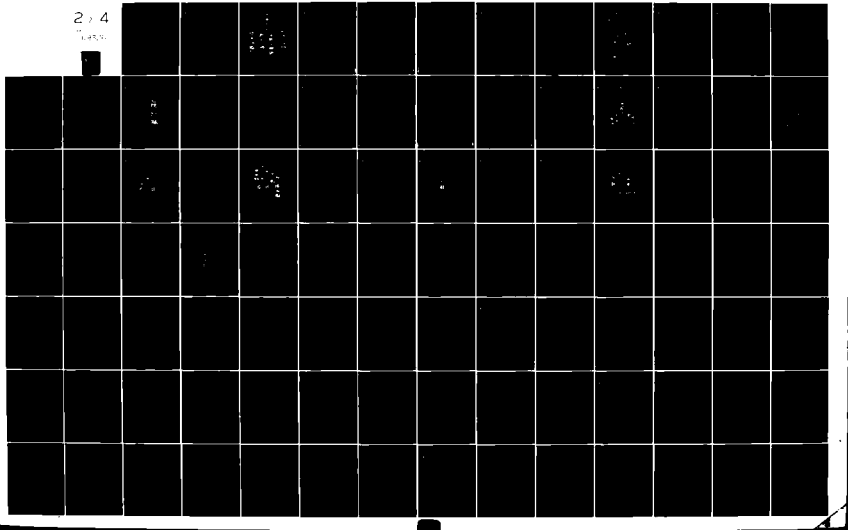
In order to introduce lateral movement between decision ellipses on different routes, pseudo-dynamic movement, the formation dictated by the route had to be overriden.[Ref. 27] When using route formations, movement is conducted by the projection of the vehicles location onto the route with respect to the movement control points (MCP) on that route. The vehicle is considered to be through an MCP as soon as its projection on the route is. When moving laterally, this does not suffice. For lateral movement, the vehicle itself and not its projection must pass through the MCP. This action is accomplished, by setting an existing vehicle attribute, FORMACODE, to zero. Setting FORMACODE to zero overrides the route formation and causes the entity to proceed directly to an MCP. This insures that the entity gets to the MCP, not just its projection.

At a decision ellipse, several additional actions take place. First a check is made to ensure that the entity is moving in the direction of the attack, rather than coming back from the obstacle. If the latter is true, the reaward

AD-A119 326 NAVAL POSTGRADUATE SCHOOL MONTEREY CA F/G 15/7
THE ENGINEER EFFECTS MODULE FOR THE STAR COMBAT MODEL.(U)
MAR 82 S C MAIN, J V MUDD

UNCLASSIFIED NL

2 > 4



movement of the entity is halted and pseudo-dynamic route selection, either to a cleared lane or a bypass route, occurs. A decision ellipse may be entered in any of three different ways,

1. Moving in the direction of the attack.
2. Returning from the obstacle, (moving opposite to the direction of attack).
3. Moving from the flank, (lateral movement from another decision ellipse).

Each direction of entry into the decision ellipse has certain tactical alternatives associated with it. The following table associates the tactical alternatives of the module listed in table 8, with the appropriate direction of entry into a decision ellipse.

Table 10: Decision Ellipse Tactical Alternatives

Direction of Movement	Alternatives from Table 8
1. In the direction of the attack	1,2, and 3
2. Returning from obstacle	3
3. From the flank of ellipse	3

The remaining internal actions are identical to those discussed in the general case in section D.3., above. At an exit from a decision ellipse, no further actions take place.

2. The Minefield Ellipse

The minefield ellipse actions are probably the most complicated of the actions to be discussed. As in the case of the decision ellipse, the minefield ellipse possesses attributes which are used to control the actions in the minefield. These attributes are listed below.

Table 11: Minefield Ellipse Attributes

Attribute name	Scatterable minefield	Pattern minefield
P1.FLD	number of mines in field	state of minefield activation
P2.FLD	trip wire length	belt separation
P3.FLD	mine type	mine type
P4.FLD	probability of detection and avoidance	
P5.FLD	probability of pushing a dead plow tank	
P6.FLD	knowledge level (0,1,2,or 3)	
P7.FLD	cleared lane formation number	
P8.FLD	cleared lane speed factor	
P9.FLD	plow activation delay time	
P10.FLD	plowing speed degradation factor	
P11.FLD	pushing speed degradation factor	
P12.FLD	minefield visual detection distance	

a. Detailed Minefield Entry Actions

Upon entry into a minefield the first action that takes place is to determine if the minefield is activated. (See Figure 12) If the minefield is not activated, the entities move through the field as if it did not exist. The entity entering the minefield is then checked to determine if is a dismounted infantryman or a vehicle. If the entry is made by a dismounted infantryman,

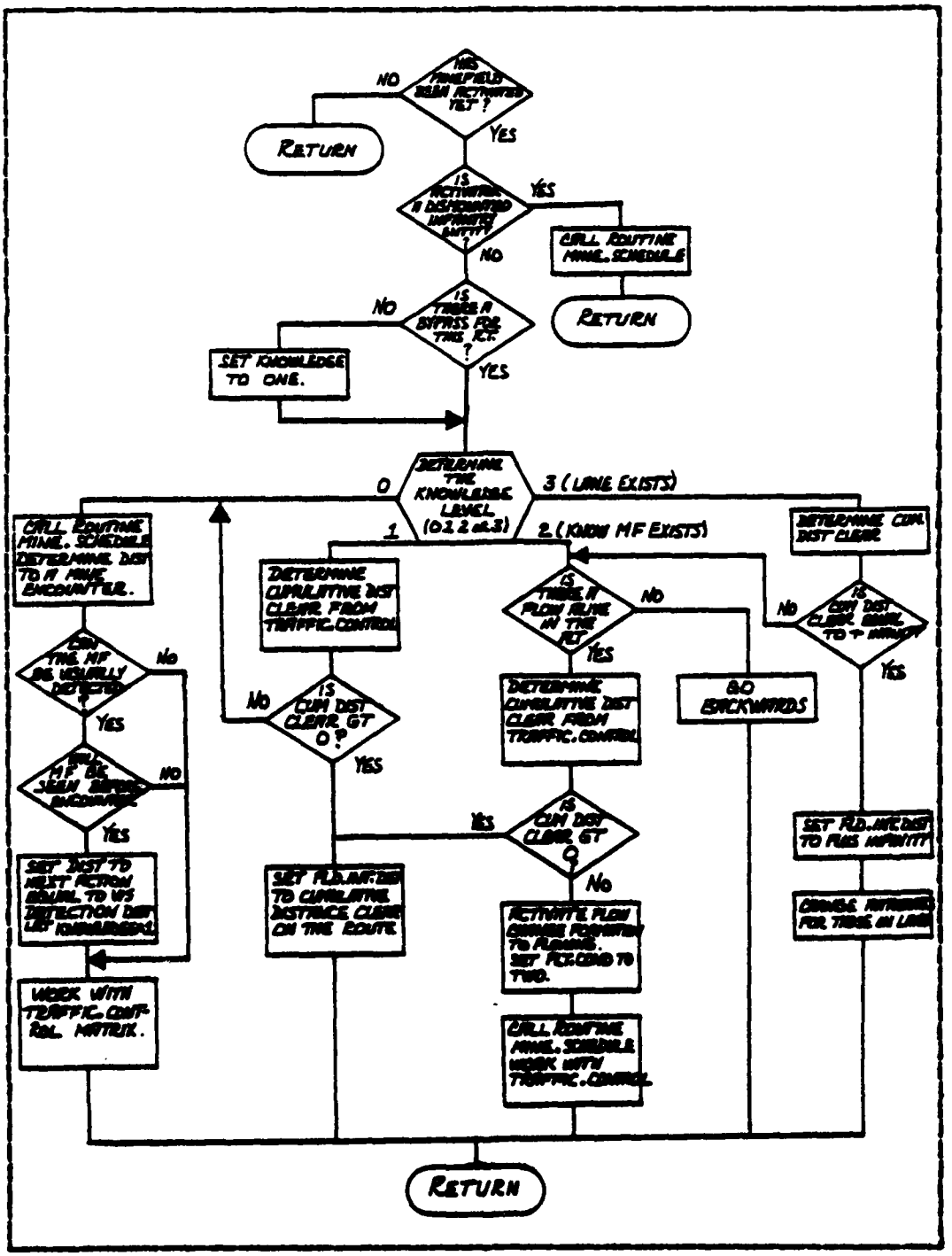


Figure 12: Minefield Entry Action Flowchart

then routine MINE.SCHEDULE is called and the actions previously discussed take place. The reason for this is that dismounted infantryman do not activate anti-vehicular mines. Immediate action drills by infantrymen are not modelled. When a vehicle enters a minefield, a check is made to ascertain if there are bypass routes for this minefield and route. If a bypass route does not exist, then the knowledge attribute is set to one.

(1) No Prior Knowledge Exists. When the knowledge level is zero, the following sequence of events takes place. Routine MINE.SCHEDULE is called and a distance to a mine encounter is determined. This becomes the field internal distance (FLD.INT.DIST). A test is then made to determine if the minefield can be visually detected. When the minefield can be visually detected, a test is made to determine if the minefield will be detected prior to travelling the distance to the mine encounter. [Ref. 25,26] When the visual detection distance is greater than zero, it is compared to the computed distance to the mine encounter (FLD.INT.DIST). The shortest distance of the two then becomes the distance to the next internal action. If the visual detection distance is the smaller distance, the

knowledge level is set to one without a mine detonation occurring. If the mine detonation distance is the smaller distance, there will be a mine detonation at the internal action and the knowledge level will be set to two. Whichever distance is selected, the route's cumulative distance clear and platoon movement number get updated in the TRAFFIC.CONTROL matrix. Logic control is then returned to the MOVE routine.

When the knowledge level is one on entry, the TRAFFIC.CONTROL matrix is checked to see if the cumulative distance clear on the route being traversed is greater than zero. If the cumulative distance clear is not greater than zero, the no knowledge path is followed. The path for the no knowledge level is described above. If the value is greater than zero, the vehicle entering the field will have its movement number changed to the platoon movement number in the TRAFFIC.CONTROL matrix and the FLD.INT.DIST is set to the cumulative distance clear on this route. Follow-on vehicles travel safely on a path as far as other vehicles have already travelled. In order to gain this and unit movement, the TRAFFIC.CONTROL matrix was created. A complete discussion of TRAFFIC.CONTROL is contained in Appendix A.

(2) Knowledge of the Minefield Exists. When the knowledge level is two, a check is made to determine if the platoon has an alive plow. When there is no plow alive, the entire platoon is turned around, to return to the decision ellipse, by switching the platoon's starting and ending areas. In addition, the FLD.INT.DIST is set to plus infinity to ensure that the field boundary will be crossed. If an alive plow exists in the platoon, then TRAFFIC.CONTROL is checked to determine if this route through the minefield has a cumulative distance cleared. If the cumulative distance clear on the route is greater than zero, then some of the route through the minefield is clear. The FLD.INT.DIST is then set to the cumulative distance cleared, the entity moves to the cleared distance, and then control is returned to the MOVE return. If the cumulative distance clear is zero, then the platoon is stopped for the time it takes to activate the plow and attain a plowing formation (the plow in front with all vehicles following directly in the plow's path). The speed of the platoon is degraded by a factor to account for a slower plowing speed and the PLATOON.COND is set to two (plowing). The plow speed degradation factor is a user specified value. Routine

MINE.SCHEDULE is called to get the distance to the next mine encounter; TRAFFIC.CONTROL is updated; and logic control is returned to the MOVE routine.

(3) A Lane Exists in the Minefield. The final path in the minefield entry case is the when the knowledge level is three. This means that at least one clear lane exists through the minefield. The first action to take place is to determine the cumulative distance clear on the route being travelled from the TRAFFIC.CONTROL matrix. If the value is plus infinity, the cleared lane is on the route the platoon is currently traversing. The FLD.INT.DIST is set to plus infinity, the formation is changed to a column, and the speed that the platoon utilizes on the cleared lane is degraded by a speed factor that the user inputs during the initial battle set up. If the cumulative distance clear is not equal to plus infinity, then the path for a knowledge level of two is followed.

This concludes the detailed description of the minefield entry actions. The next section is a description of the minefield internal actions. (See Figure 13)

b. Detailed Minefield Internal Actions

In the MOVE routine, the FLD.INT.DIST values are continuously checked for each moving entity. Once the FIELD.INT.DIST is zero, then an internal action is triggered. The first test is to determine if the entity triggering the internal action is a dismounted infantryman or a vehicle. If the entity is a dismounted infantryman, then routine POP.A.MINE is called and the damage status of the entity is determined. When the entity is still alive, routine MINE.SCHEDULE is called, a new internal distance is computed, and control returns to the MOVE routine. When the entity is not alive, control is returned to the MOVE routine and no other actions take place. If, on the other hand, the activating entity is a vehicle, then the knowledge level of the minefield is checked.

(1) No Knowledge Exists (0 or 1). If the knowledge level is zero, then the first action is to call Routine POP.A.MINE. In this routine, the damage caused by the mine detonation is assessed. The knowledge level on the field is changed to two. If the knowledge level is one, a visually detected minefield, then routine POP.A.MINE need not be called in order to set the knowledge level to two.

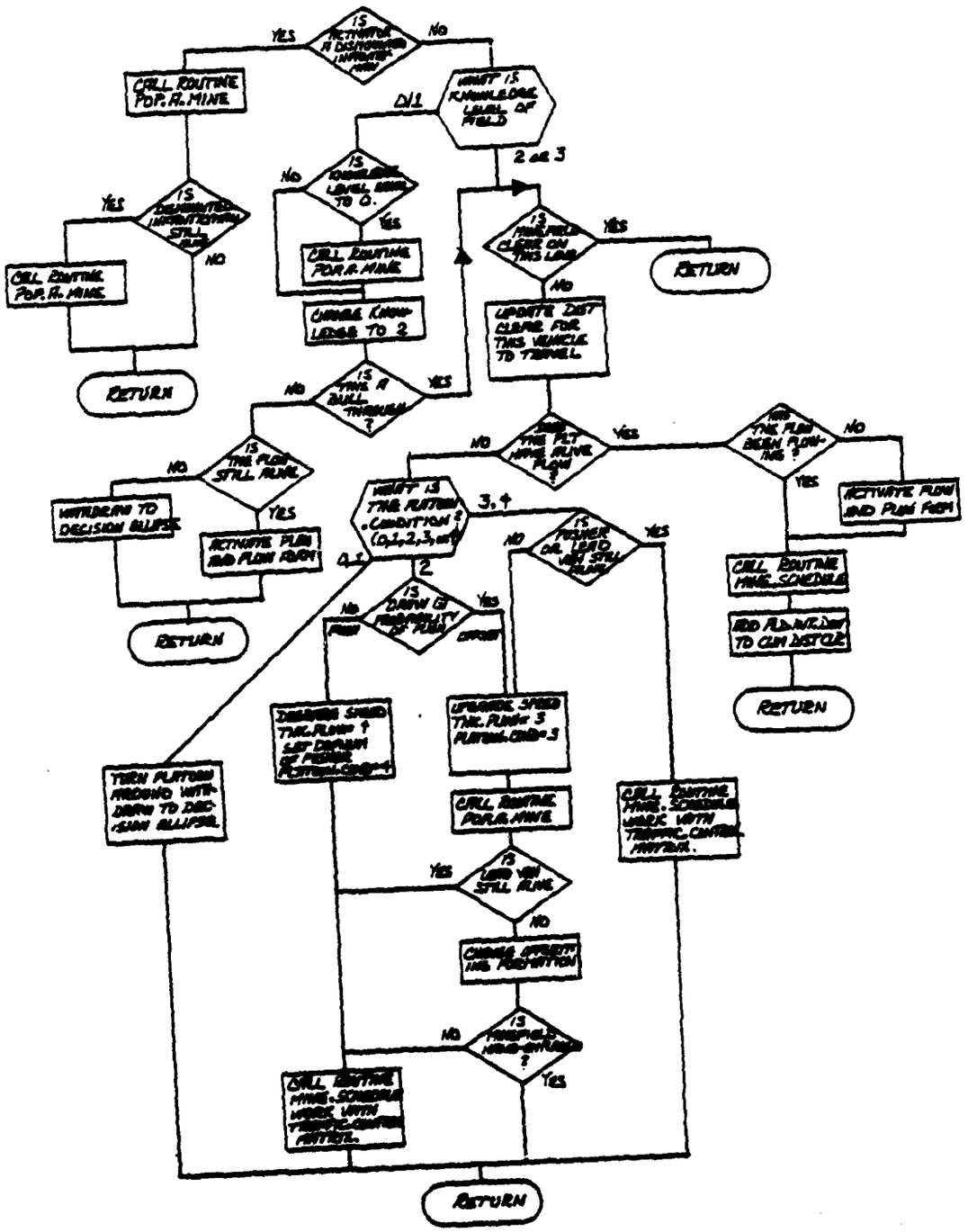


Figure 13: Minefield Internal Actions Flowchart

The next check is to ascertain if the unit is utilizing the "bull through" tactic. If the action is not a "bull through", then the next check is to determine if a plow is alive in the platoon. If a plow is alive, it is activated, the plow formation is set, and control is returned to the MOVE routine. In the case where no plow remains alive in the platoon, then the platoon is turned around and sent back to the decision ellipse. If the unit is utilizing a "bull through" tactic, the platoon follows the path for a knowledge level of two. This ends the description for the instance of knowledge levels of zero or one.

(2) Knowledge Exists(2 or 3). When knowledge exists about the minefield, the first check is to determine if a clear lane exists on the present route. If a lane exists, then the FLD.INT.DIST is set to plus infinity, the formation is changed to a column, the speed is degraded, and control is returned to the MOVE routine. If the entire route is not clear, then the distance clear to travel (FLD.INT.DIST) is updated to the latest cumulative distance clear on the route listed in the TRAFFIC.CONTROL matrix. Next, a test is made to determine if a plow is still alive in the platoon. If there is a plow still alive, then the

PLATOON.COND is checked to see if the plow is already plowing. If it is not already plowing, then the actions to get the platoon into a plowing configuration are called and a hasty breach is begun. If the plow is already plowing, then routine MINE.SCHEDULE is called yielding a new distance to a mine encounter and the TRAFFIC.CONTROL matrix is updated. Control is now returned to the MOVE routine. If there is no plow still alive in the platoon, then the platoon plowing condition serves as the next branch point for our actions. To review the situation, the platoon is in a minefield, has knowledge of the minefield, and there are no plows alive in the platoon. If the PLATOON.COND is equal to one, the platoon had a plow upon entry into the minefield, but had never employed it. At this point, the platoon is sent back to the decision ellipse and control returns to the MOVE routine. If the PLATOON.COND is two, then the platoon had been plowing already. This means that the platoon has two courses of action available to it, it can elect to push the dead plow, or offset around it with the remaining live vehicles. The choice between the two options is a probability that is specified by the user. A random number is drawn from a Uniform (0,1) distribution and

Monte Carlo'ed to determine which choice will be taken. If the pushing option is selected, then the actions that take place are:

1. The platoon speed is degraded to the speed of a tank being pushed.
2. The PLOW.COND of the push tank is set to four in order distinguish the push tank from the other vehicles of the platoon. The PLATOON.COND is set to four, indicating that the platoon is utilizing the pushing option.
3. The defilade number (DEFNUM) of the pushing tank is changed. This change affords the pusher partial cover from direct fire.
4. Routine MINE.SCHEDULE is called and the TRAFFIC.CONTROL matrix is updated.

The selection of the offset option would trigger the following actions:

1. The speed of the platoon would be upgraded back to the maximum speed allowed by the terrain.
2. The activating vehicle has its PLOW.COND set to three, indicating that it is the lead vehicle of the platoon. The PLATOON.COND is also set to three indicating that the platoon is in the offset sequence.

3. Routine POP.A.MINE is called and damage to the lead vehicle is assessed.
4. If the lead vehicle is still alive, then routine MINE.SCHEDULE is called, TRAFFIC.CONTROL is updated, and control is returned to routine MOVE.
5. The formation of the platoon is changed to an offset formation and the mine type is checked. The mine type is checked because in a pattern minefield, offsetting vehicles are allowed to hit the same belt that killed the lead vehicle. If the type is scatterable, it is an expected distance that is computed for the offsetting tank to move prior to a mine encounter.

The final check at this branch is for a PLATOON.COND of three or four. This means that the platoon had already been pushing or offsetting. The status of the lead tank or the push tank is checked. If this tank is still alive, then routine MINE.SCHEDULE is called to determine the distance to the next mine encounter and the TRAFFIC.CONTROL matrix is updated. If this vehicle is dead, then the offset option is chosen. It is felt that the task of pushing one dead tank that has a plow on it will be very difficult; the task of pushing two dead tanks, impossible.

Thus, from this point on, the remaining vehicles in the platoon will have to offset around any dead vehicles. Their actions will follow the offset case previously described.

This completes the detailed description of the minefield internal actions. The final point of discussion in this chapter is actions to be taken at an exit from a minefield.

c. Detailed Minefield Exit Actions

The actions that take place upon exit from a minefield are relatively easy to understand when one compares them to the actions discussed already. The reference for this discussion is Figure 14.

The first action to take place on exit is to upgrade the vehicle speeds to the maximum allowed by the terrain. The second is to determine if the entity is exiting on the way back to a decision ellipse associated with this minefield, or moving in the direction of the attack. If the movement is in the direction of the attack, then a lane has been cleared for other units to follow. The following values are then set:

1. The knowledge level on the the field is set to three. (a lane exists)
2. In the TRAFFIC.CONTROL matrix the cumulative distance cleared for the route is set to plus infinity.

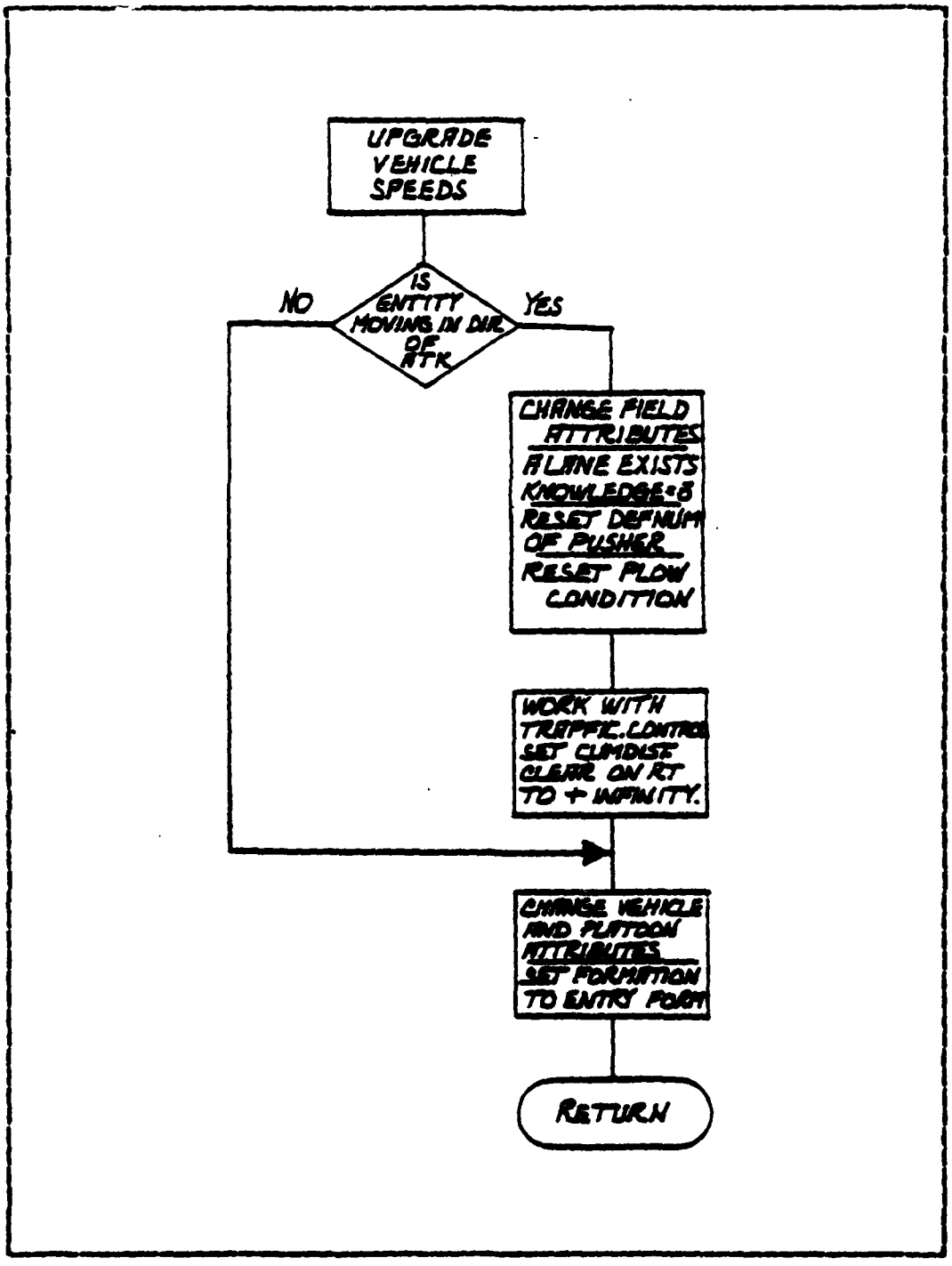


Figure 14: Minefield Exit Actions Flowchart

3. The platoon has its formation reset to the formation directed by this route.

4. The vehicle attributes pertaining to minefield actions are reset so that future minefield encounters will not be affected by this breach. Control is returned to the MOVE routine.

If the entity is moving back to a decision ellipse on exit, then the platoon has its formation reset to the one used on entry and continues its movement.

F. CONCLUSION OF THE MINEFIELD MODEL

This concludes the discussion of the model itself and how the actions take place. The serious user of this model should be able to take this description, the STAR running instructions, and the SIMSCRIPT II.5 code included in the appendices and have minefields and their synergistic effects in a STAR Combat simulation run. In the next chapter, engineer gap obstacles, such as, the tank ditch, road crater, and blown bridge (short-wet gap), are discussed. The discussion of the modelling of these obstacles will generally follow the same format as the description in this chapter.

VI. THE ENGINEER GAP OBSTACLES

A. INTRODUCTION

This chapter discusses the simulation of the Engineer gap obstacles in the STAR Combat Model. As in the case of the minefield, first the obstacles must be modelled and then the effects of the obstacles on the combat process can be simulated. The gap obstacle portion of the Engineer Effects Module allows obstacles to be emplaced, impede the attacker and cause the attacker to execute predetermined tactical options. Several of the new methods and terminologies introduced in the minefield chapter are also used with the gap obstacles. Consequently, this chapter requires a somewhat less detailed explanation than previous discussions.

B. GAP OBSTACLES MODELLED

In order to model the effects of the gap-type obstacles on the combat process, it was first necessary to install the gap obstacles in the STAR model. The Engineer gap obstacles that are modelled and described in this chapter include, the tank ditch, road crater, and blown bridge (short-wet gap).

1. The Tank Ditch

The tank ditch is a linear obstacle that is modelled as a path 3.3 meters wide, corresponding to the major axis of a field ellipse. The size of the ellipse is determined by two factors, the actual length of the tank ditch and its visual recognition distance. The length of the semi-major axis of the ellipse is equal to half the length of the tank ditch, and the semi-minor axis is the visual recognition distance. (See figure 15)

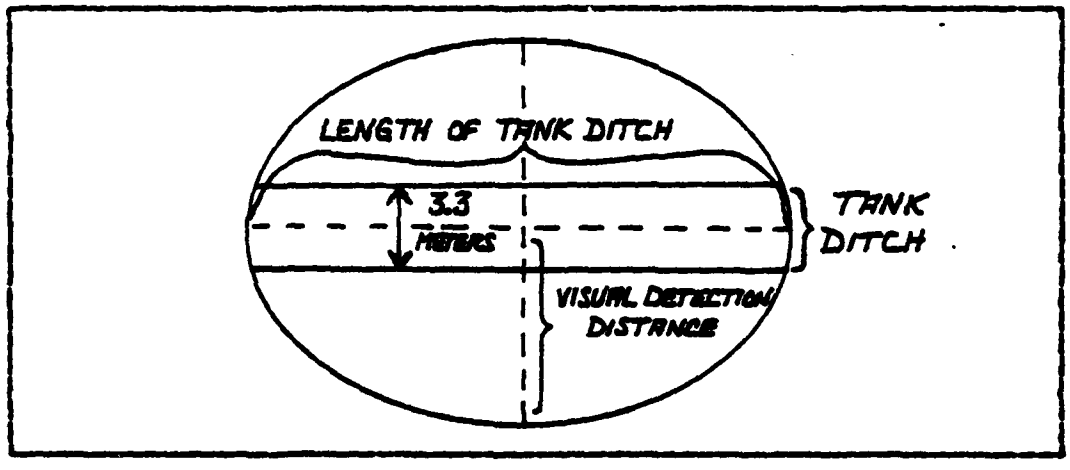


Figure 15: Tank ditch representation

2. The Road Crater

The road crater is a point type obstacle, represented as the center of a field ellipse, a circle

located on a road or route. The radius of the circle is the visual recognition distance. These craters (circles) are placed on the suspected routes of the attacking force and are designed to delay and disrupt the channelized movement of this force. (See figure 16)

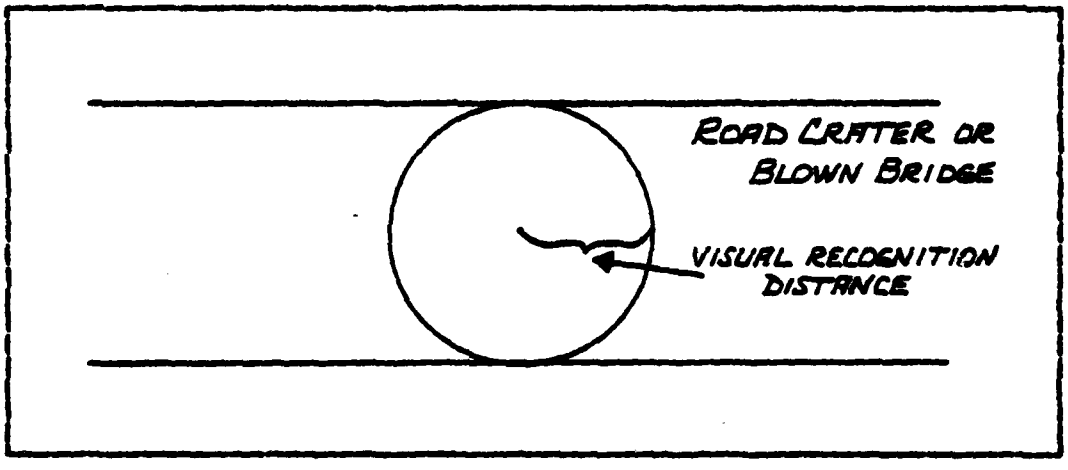


Figure 16: Road Crater/Short-Wet Gap Representation

3. <u>The Blown Bridge (short-wet gap)</u>

The short wet gap is represented in a manner that is very similar to the road crater. The field ellipse representing the short-wet gap is constructed so that the route or road is covered by the circle at the desired point. (See figure 16) Vehicles cannot proceed past this point without bridging assets. In this model, swimming or snorkeling of vehicles across the wet gap is not simulated.

C. THE GAP OBSTACLE IN STAR

The simulation of the gap obstacle in STAR produces certain synergistic effects, one such effect is the selection of tactics to overcome/neutralize enemy obstacles. In Chapter five, three minefield tactical alternatives for the commander were specified. These same three alternatives are appropriate for gap type obstacles and are also modelled. For the reader's convenience, the alternatives are listed below.

Breach the obstacle

Bypass the obstacle

"Bull Through" the obstacle

As in the minefield model, the control mechanism for the implementation of the above three alternatives is the decision ellipse. The status of a unit's organic breaching equipment, however, still serves as the major factor in the selection of the alternative tactical responses. The types of breaching equipment are different but the concept remains the same. In addition, some breaching equipment is only effective against specific gap obstacles. For example, a dozer blade equipped tank cannot breach the short-wet gap, however, it can be used to fill the tank ditch or road

crater with earth to allow passage of vehicular traffic. The Armored Vehicle Launched Bridge (AVLB), on the other hand, is effective against all the gap obstacles, since it is emplaced across the gaps created by the obstacles. Finally, the self-breach is akin to the "bull through" instance from the minefield model. This means that the vehicle can cross some obstacles by itself. This is accomplished in a rather crude fashion; the tracked vehicle oscillates back and forth in the ditch or crater, trying to cave the sides of the obstacle walls in to allow for an easier exit of the vehicle. This exposes the vehicle to the high risk of becoming immobilized in the obstacle because of mechanical failures. This method is also very costly in terms of the amount of time that is lost while conducting the self-breach. Only the road crater or the tank ditch may be self breached.

Now that the gap type obstacle representation has been briefly outlined, it is time to discuss the synergistic effects created by the emplacement of these obstacles on the battlefield. As in the previous discussion of the minefields, the actions will be discussed with respect to flowcharts included with the text. (See Figure 17)

1. Gap Decision Ellipse Actions

As the reader may recall, a decision ellipse can be entered in any of three ways, toward the attack on a route, coming back from an obstacle and from the side, when an entity is moving laterally in search of a bypass route.

An entity entering a gap decision ellipse from the flank has its attribute (FORMACODE), that initiated the lateral movement, reset to allow for future dynamic route selections. If this decision ellipse is on a route that circumvents the obstacle in question, then entities entering this particular decision ellipse are given the formation dictated by the bypass route and sent on this route toward their objective. Dismounted soldiers are not affected by gap obstacles and are allowed to proceed directly through them. If the entity enters the decision ellipse on the way back from an uneventful breaching attempt, then his rearward movement is altered so that pseudo dynamic route selection (bypass/lane) may occur. At this juncture, the knowledge link to the obstacle is performed for all entering entities regardless of the direction of travel upon entry.

Prior to continuing, a brief review of the order of preference for tactical responses will be listed. If a unit

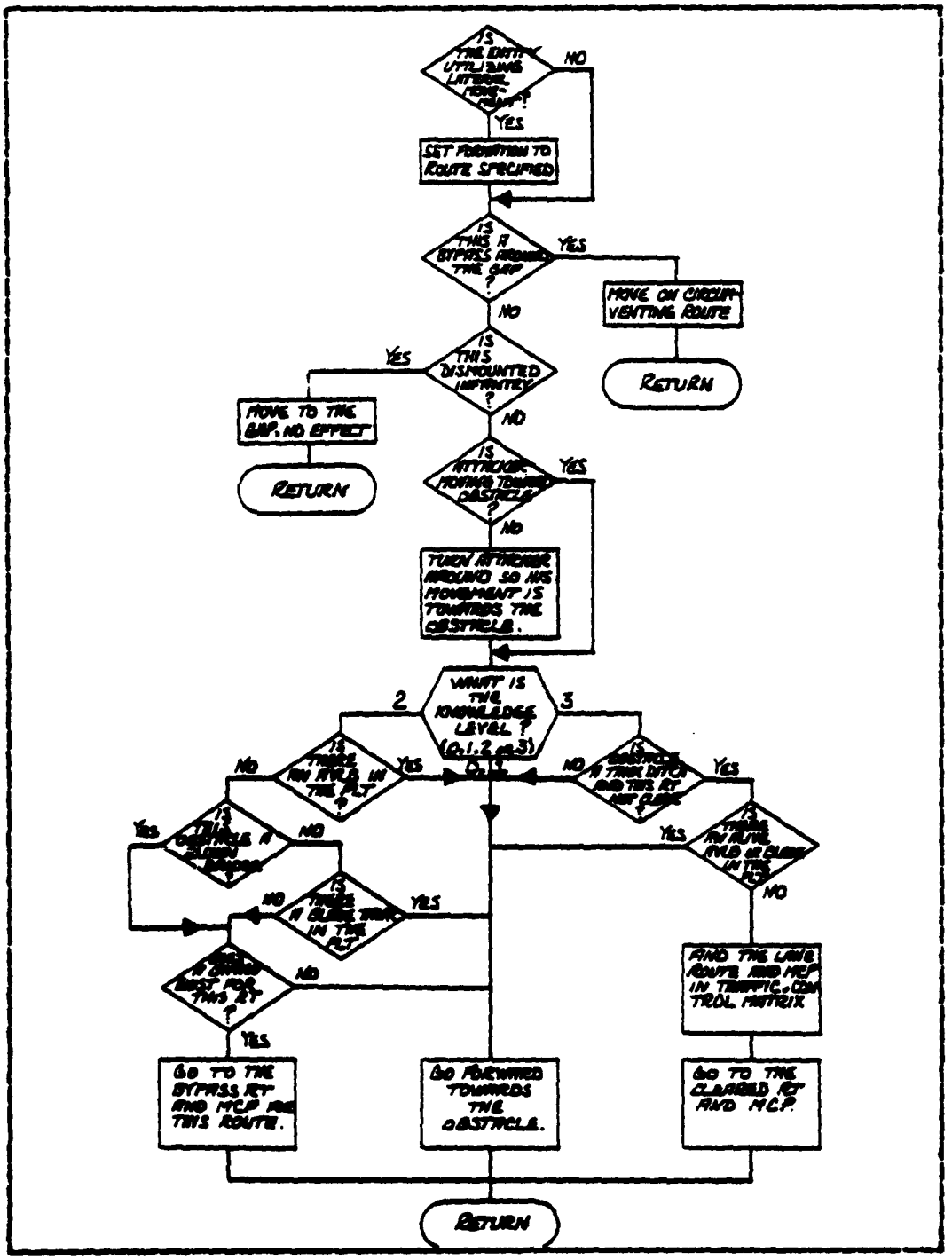


Figure 17: Routine GAP.DECISION Flowchart

possesses the appropriate breaching equipment to counter an obstacle, then it will always attempt to breach the obstacle. If this is not the case and the "bull through" is not specified by the user on input, then a lane select or a bypass option will occur.

When an entity enters a decision ellipse and has no knowledge of the obstacle's existence, then the entity is sent directly toward the obstacle to continue the attack. If, however, the unit is aware of the obstacle's existence, (the knowledge level is two), then the unit is checked for possession of the appropriate breaching equipment to counter the impending gap type obstacle. The application of a particular piece of breaching equipment against a particular obstacle was discussed earlier in this section. If the platoon possesses breaching equipment, then it is allowed to continue in the attack toward the obstacle. The next check made when the knowledge level is two is to determine if the unit will bypass or "bull through" the obstacle. This determination is necessary only when the unit does not have applicable, organic breaching assets to neutralize the obstacle to its front. If the "bull through"/self breach is specified on input, then the entity continues to move toward

the obstacle. If the bypass option is chosen, then the unit will obtain the Bypass Route and Movement Control Point from the field's attributes and then move to the Bypass Route and MCP. If information about a breached lane through an obstacle exists, (knowledge level of three), then a platoon without proper breaching equipment will proceed to the lane that already exists through the gap. If a unit has the necessary equipment tc perform a successful breach, it will always attempt to breach a lane on the present route of movement.

The Decision Ellipse is the main control mechanism for the selection of tactical alternatives available to the unit commander. The next section of the gap barrier discussion pertains to actions at and in the gap obstacle field ellipse. (See Figure 18)

2. Gap Entry Actions

The first check made when an entity enters a gap obstacle is to determine if the obstacle is emplaced/active. If the obstacle is not active, then the entering entity continues its unaffected movement in the current direction. At this point, a check to determine if the entity is mounted is made. If the entity is not in a vehicle, then it is

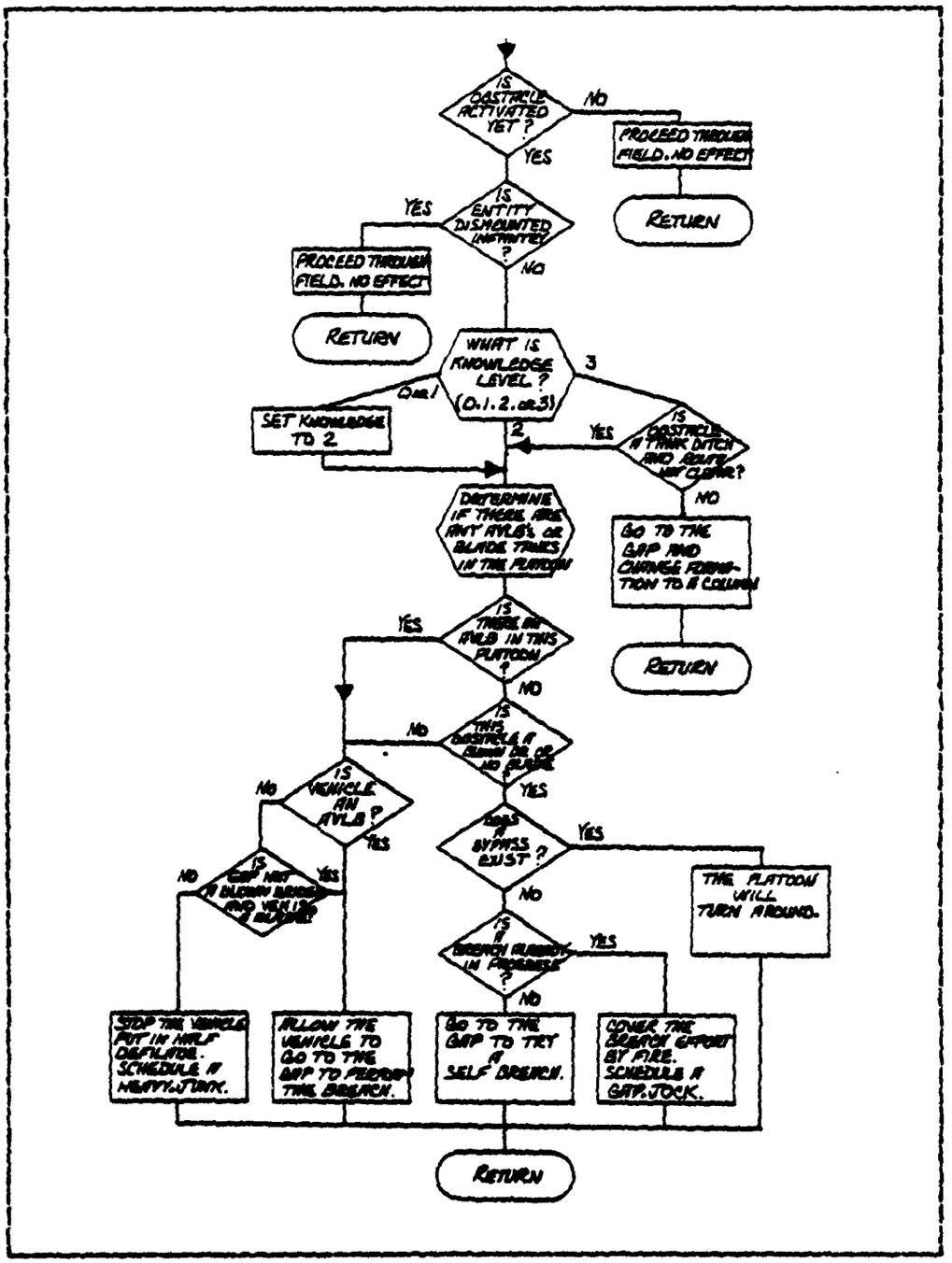


Figure 18: Routine GAP.ENTRY Flowchart

allowed to proceed unscathed through the barrier. Again, dismounted entities are assumed to be unaffected by gap type obstacles. Upon completion of these initial checks, the knowledge level of the obstacle is ascertained. If the knowledge of the obstacle is nonexistent, then the level is now switched to two, the entity and all affected units now know the location of this obstacle. The reader is reminded that the gap type obstacle field perimeter is defined by the visual recognition distance to the obstacle. If the route of present travel is a cleared/breached lane across the gap, then the entity will change formation to a column and proceed to the crossing site. If this route does not contain a breach, then a check of the status of the platoon's organic, breaching equipment is made. If the platoon possesses appropriate breaching equipment to counteract the obstacle, then the equipment is sent forward to gain a breach across the gap. The ongoing breach flag, located in array TRAF.CONTROL is set to the number of the platoon presently attempting the breach on this route (see Appendix A for more details). The remainder of the platoon is halted, placed in half defilade positions, and provides overwatching fire for the breaching operation. An event is

scheduled every fifteen seconds to update the overwatching entities on the progress of their platoon's breaching operation. This event is called HEAVY.JUNK and is discussed in detail in Section five, below. When a platoon possesses no organic breaching equipment, a test is made to determine if a bypass route exists for the entities on this route. If the unit can bypass, then the entity's platoon is turned around and sent back to the decision ellipse. If the unit cannot bypass, then it must "bull through". When this is the situation, a check is made to determine if a breaching operation is already underway on this route. If a breaching attempt is already in progress, the entity's platoon will stop, move into half defilade positions, and cover the other platoon's breaching operation by direct fire. When this occurrs an event called GAP.JOCK is scheduled. (See Figure 19) If, however, no breach is underway, then the entity moves to the gap and attempts a self-breach.

3. Event GAP.JOCK

This event simulates a breaching operation status report for the overwatching fire entities. If the breaching operation is still in progress and the breaching/ bridging equipment is alive, then the supporting entities continue

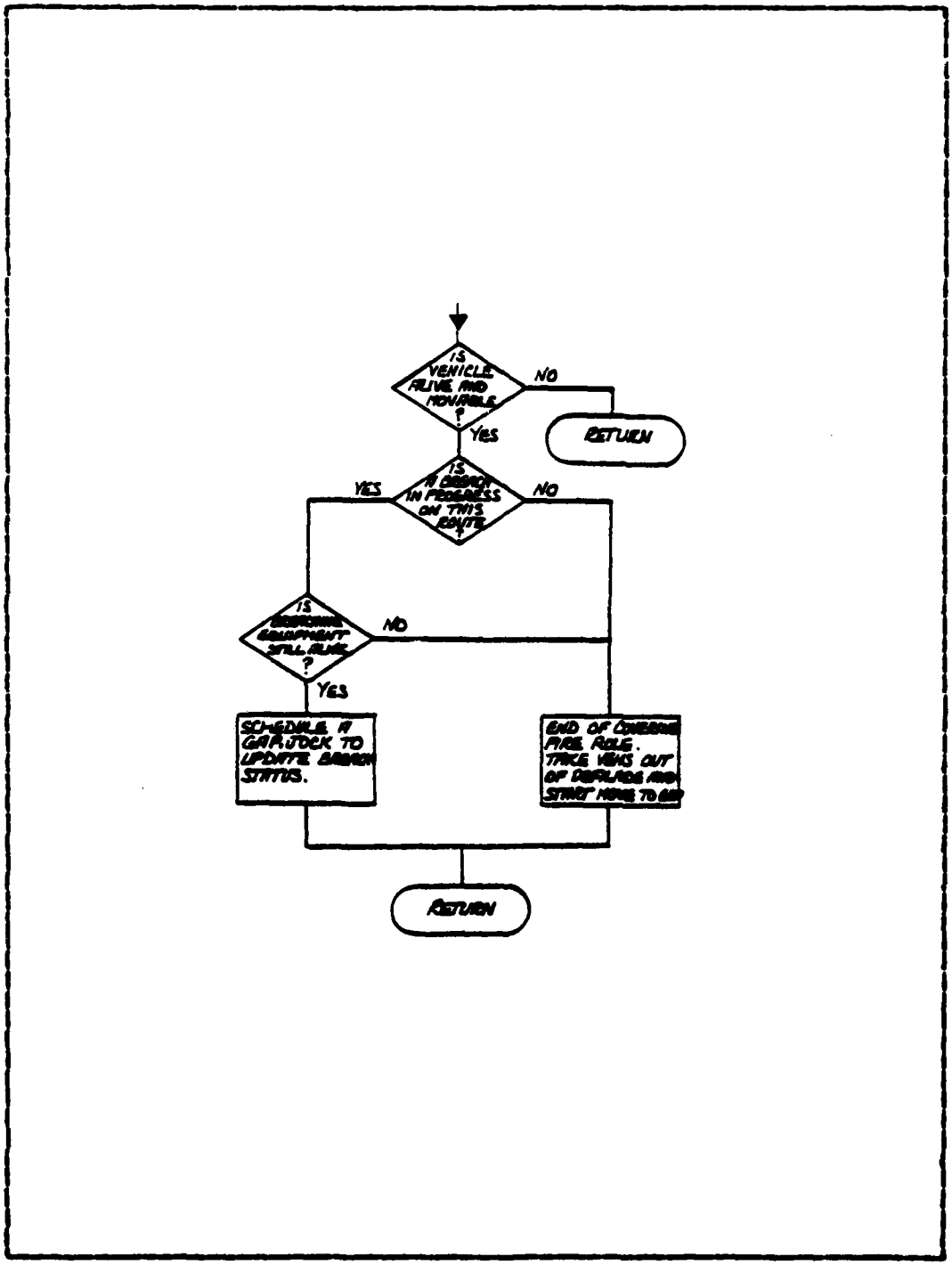


Figure 19: Event GAP.JOCK Flowchart

their covering fire role. If, however, the operation has stopped, regardless of success or failure, the checking entity starts moving toward the gap to either cross over the successful breach or attempt a self-breach.

This completes the discussion of the gap obstacle field entry actions and the gap internal actions will now be explained. (See Figure 20)

4. Gap Internal Actions

At the gap, the knowledge level of the obstacle can only have two possible values, two or three. The first, representing an obstacle without a breach and the second, an obstacle with a breach/span across the gap. If the obstacle is spanned/breached (knowledge level of three), and the route being travelled is the cleared path, then the entity changes its formation to a column and reduces its speed in order to move through/over the crossing site. When no breach exists, then the breaching equipment, the Blade Tank and AVLB, must be programmed to perform different actions upon arrival at the obstacle than the self-breach type of tracked vehicles.

In order to make the distinction between the types of breaching equipment, two checks are made. These checks

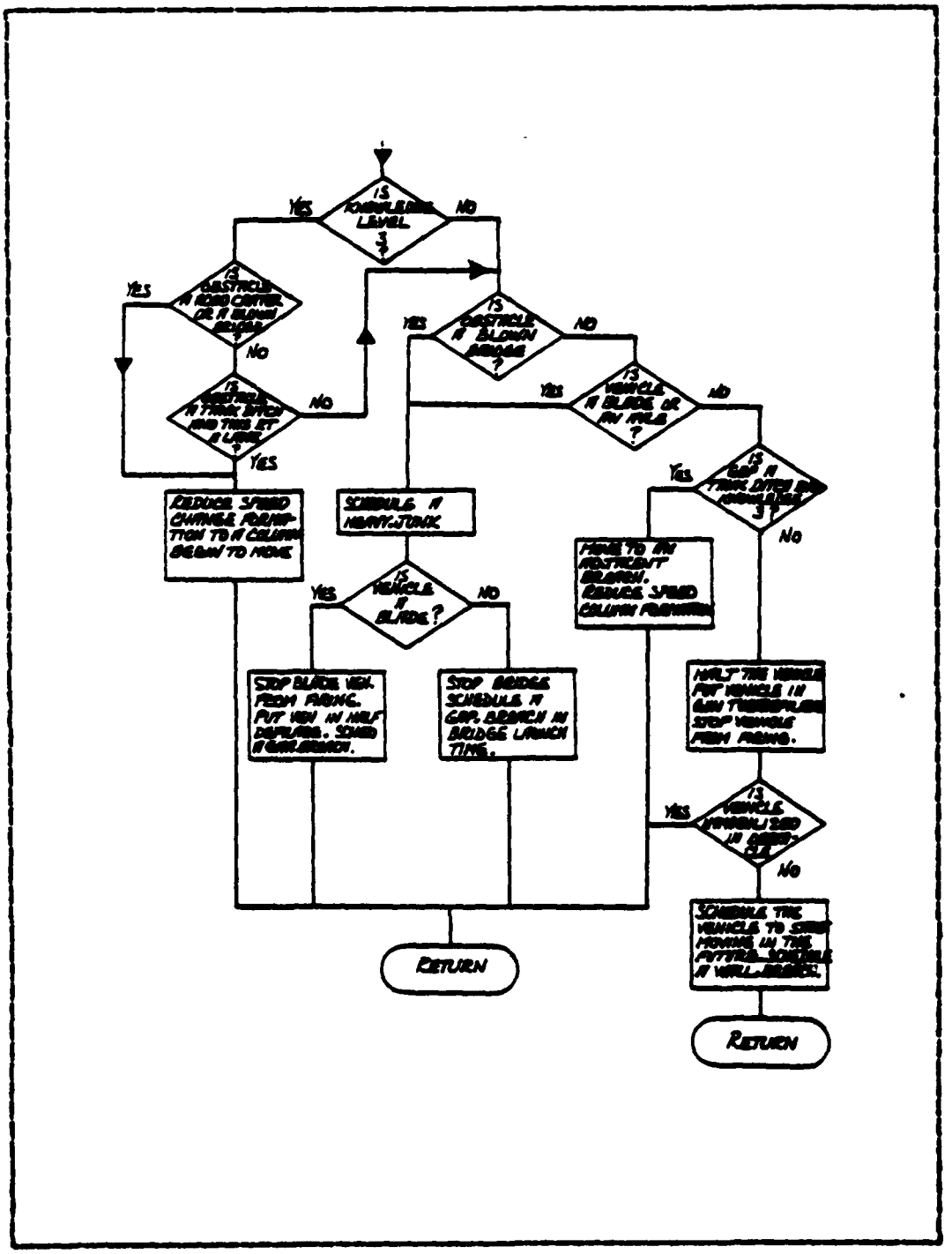


Figure 20: Routine GAP.INTERNAL Flowchart

allow the appropriate actions to take place depending on the type of obstacle encountered and the type of equipment available in the platoon encountering the obstacle. In making the distinction between the two categories of potential breachers, when there is no breach through the obstacle on the route of travel, a check is made to determine if the obstacle is a destroyed bridge (short-wet gap). This check is made because the only entity allowed to reach a blown bridge location that has not been spanned, is an AVLB. The Blown Bridge obstacle always has a bypass option; the "bull through" is not permitted. If the obstacle is a road crater or tank ditch, then a check is made to determine if the entity causing this internal action is a piece of breaching equipment, a dozer blade equipped tank or an AVLB. When the activating entity is not a breaching vehicle, a check is made to determine if the obstacle is a tank ditch and the knowledge level is three. If this is the case, then the entity is simulated as moving to the adjacent breach by drastically reducing its speed and directing this platoon element into a column formation. This speed and column change feigns this lateral movement; the vehicle actually stays on its current route. If the

obstacle had no breach, (knowledge level of two), then the entity must self-breach. In this situation the vehicle is halted, put in gun tube defilade to simulate being in or at the bottom of the ditch, and is prevented from firing due to its position in the ditch. The probability of immobilization during a self breach is Monte Carlo'ed to determine if the entity will become a mobility casualty in the future. If the entity was fortunate enough to avoid becoming stuck in the ditch, then the vehicle is scheduled to start to move after sufficient time has elapsed to simulate a gap penetration by self breaching. If the obstacle is breached in this manner, then an event WALL.BREACH is scheduled to occur after the alloted time has passed and the vehicle is taken out of its defilade posture and allowed to fire again. The purpose of WALL.BREACH is to update the knowledge level of the obstacle to three in order to indicate this route as a breached lane through the obstacle to follow-on units. (see figure 21)

This completes the description of the internal action with a breach not in existence on the route of travel and the platoon void of any breaching equipment.

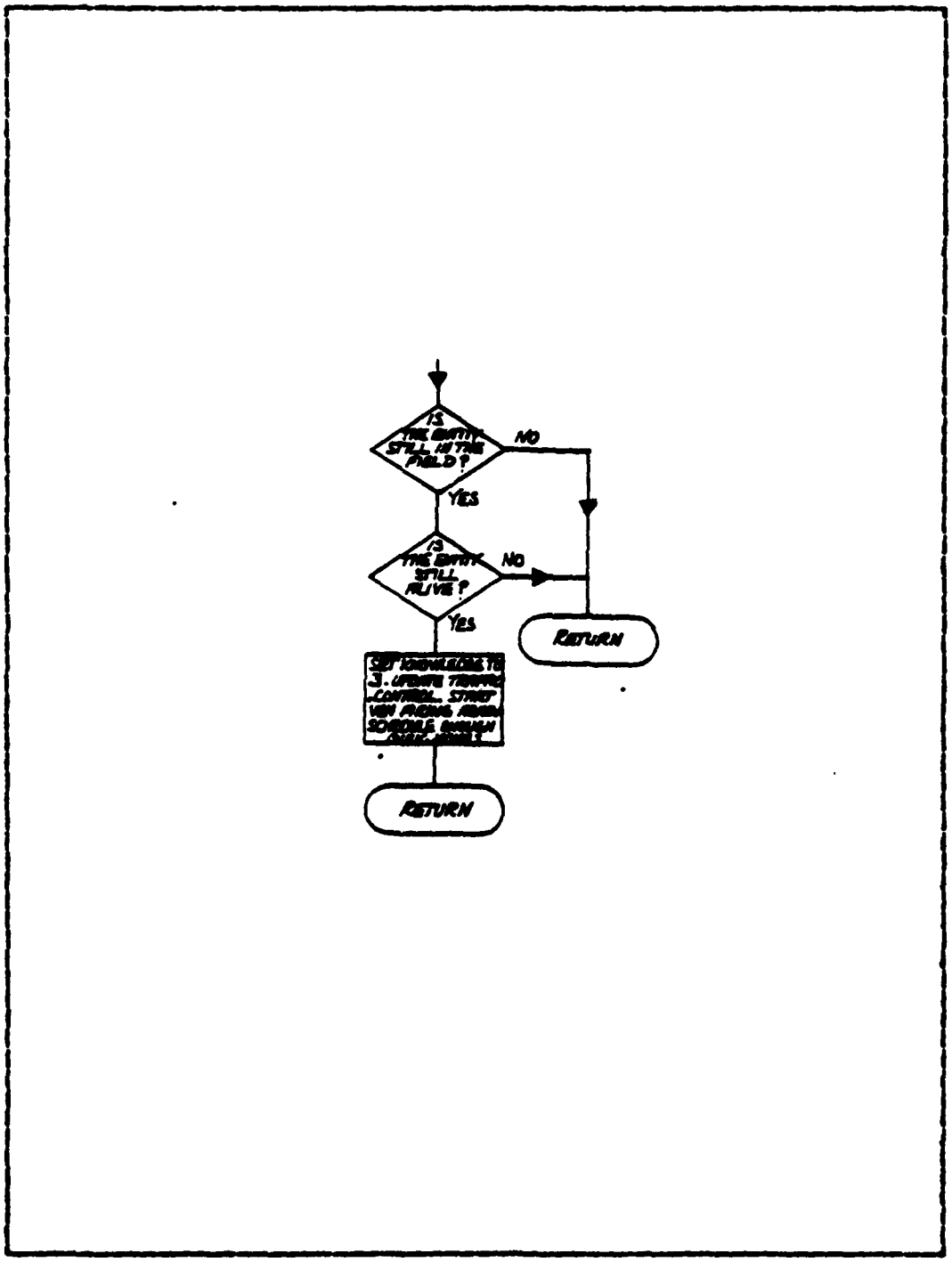


Figure 21: Event WALL.BREACH Flowchart

The discussion continues with the situation when the knowledge level is two and the platoon has breaching equipment on hand. For this discussion, the reader is again directed to figure 20. If the vehicle causing this internal action is a dozer blade equipped tank, then the tank is halted, put into half defilade while it fills in the excavation, and its direct fire capability is removed. When the breaching device is an AVLB, the bridge vehicle is stopped on the friendly side of the gap. In both cases an event GAP.BREACH is scheduled to occur after sufficient time has elapsed to simulate the appropriate breach/span. This event will be discussed in section 6 below. The possibility exists that an earlier breach was scheduled for this route. In order to simulate this situation, an event called HEAVY.JUNK is scheduled to update the breaching progress status of the obstacle.

The discussion has covered all the actions that can take place at the gap. At this point two events, HEAVY.JUNK and GAP.BREACH have been scheduled. These events and their ramifications with respect to the simulation will now be explained. (See Figure 22)

5. Event HEAVY.JUNK

Event HEAVY.JUNK is scheduled by the platoon overwatching elements at field entry and again scheduled by the breaching equipment at the gap. This event serves as a situation report on the progress of the breaching effort. If the breaching effort is continuing, the platoon breaching equipment is checked. If the appropriate breaching equipment to counter the obstacle in question is still alive, then another status report, HEAVY.JUNK, is scheduled to take place in fifteen seconds. If, however, the obstacle is not breached and the platoon effort has been exhausted, then the array TRAP.CONTROL (see Appendix A) is checked to determine if the platoon can bypass this obstacle. If the platoon can bypass the gap, then the remaining unit members are turned around and sent back to the decision ellipse where additional platoon movement actions will take place. However, if the platoon cannot bypass, then the elements have to attempt a self-breach of the obstacle. The platoon elements covering the breach are taken out of their half defilade positions and moved toward the gap where the self breach will occur.

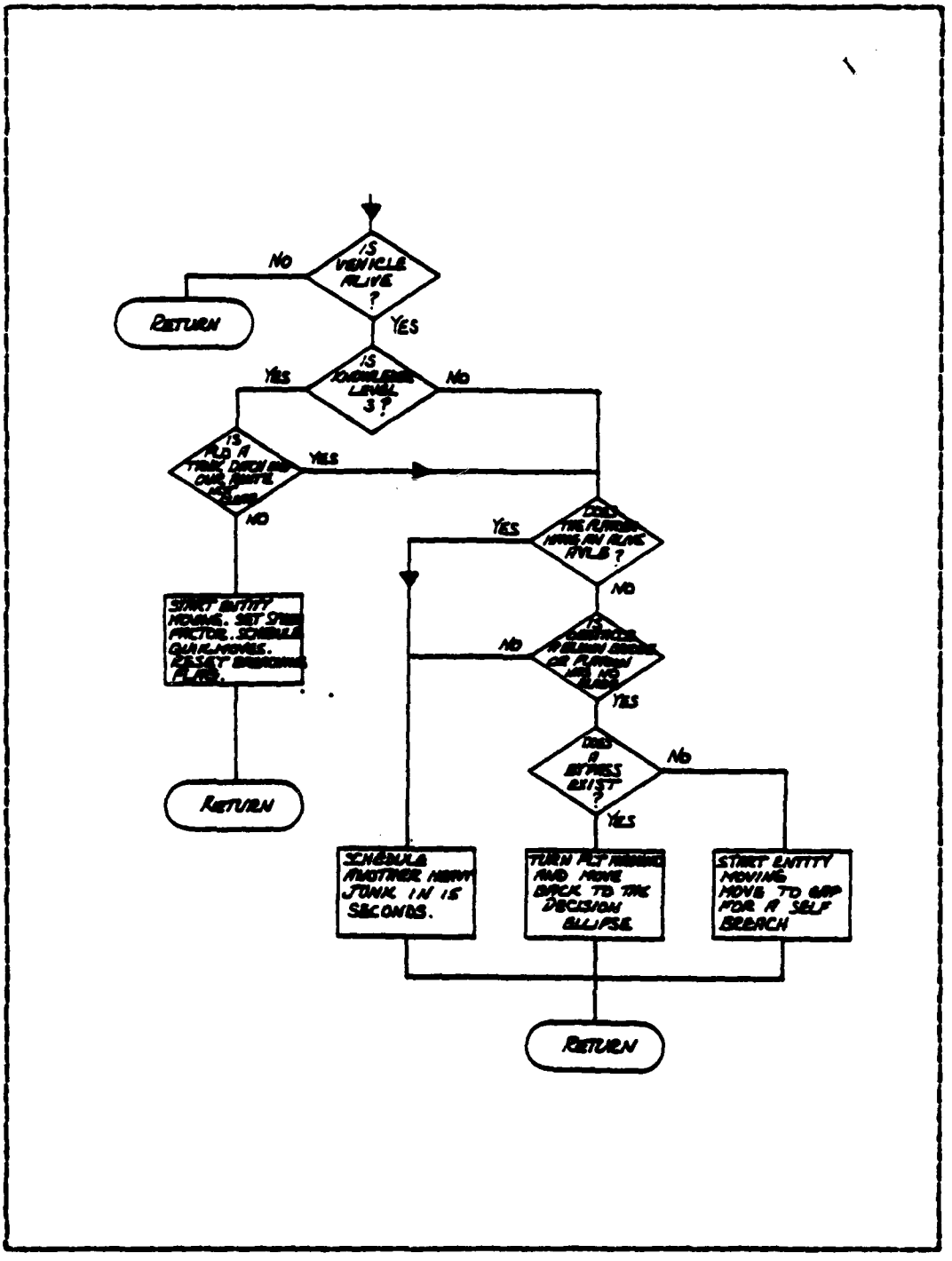


Figure 22: Event HEAVY.JUNK Flowchart

The remaining segment of Event HEAVY.JUNK pertains to the breaching equipment attempting to breach the gap. If the knowledge level is three, a breach is completed. If this route contains the breached lane, then the platoon is restarted. These entities are taken out of any defilade positions, assembled into platoon column and move in the direction of the attack. The breaching equipment departing the gap is slowed down just enough to let the remaining platoon elements catch up. During Event HEAVY.JUNK, anytime movement is initiated toward the gap, the TRAF.CONTROL flag signifying a breach in progress is turned off by setting the value to zero. (see Appendix A)

The last series of checks and actions for breaching equipment at the obstacle requires additional explanation. This is accomplished by an example. (See Figure 23)

At time 9:00, Platoon A arrived at the tank ditch in question and began a breaching operation. Two tanks are covering by fire and the platoon blade tank is starting to fill in the gap. The successful breach is scheduled to occur in six minutes, at 9:06. Platoon B is approaching the obstacle. (See Figure 24)

At 9:02 Platocn B arrives at the obstacle on the same route and schedules a breach to occur for its blade tank at 9:08.

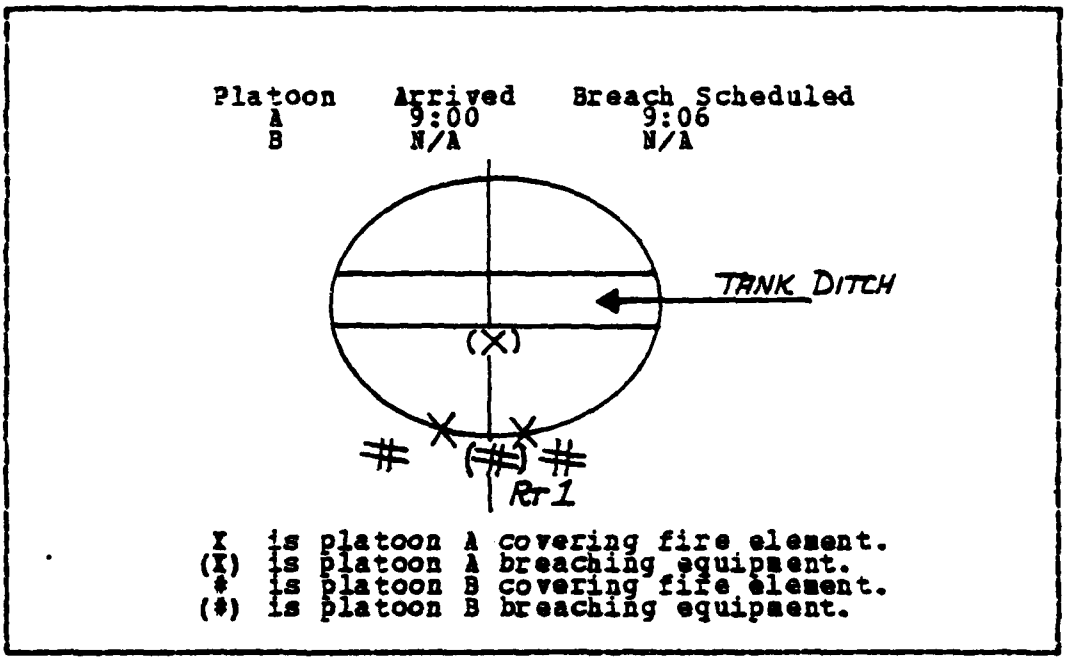


Figure 23: Multiple Breach of a Tank Ditch, Time: 9:00

Platoon B's two other tanks remain at the field boundary and cover the breaching operation by direct fire.

At time 9:06, Platoon A completes a successful breach. Route 1 is now a cleared lane and the elements of Platoon A begin to move to cross the gap. As a result of the successful breaching operation by Platoon A, it is not

necessary for Platoon B to continue its breaching operation. Through the use of the event HEAVY.JUNK, Platoon B is also started up and allowed to cross the obstacle at approximately 9:06.

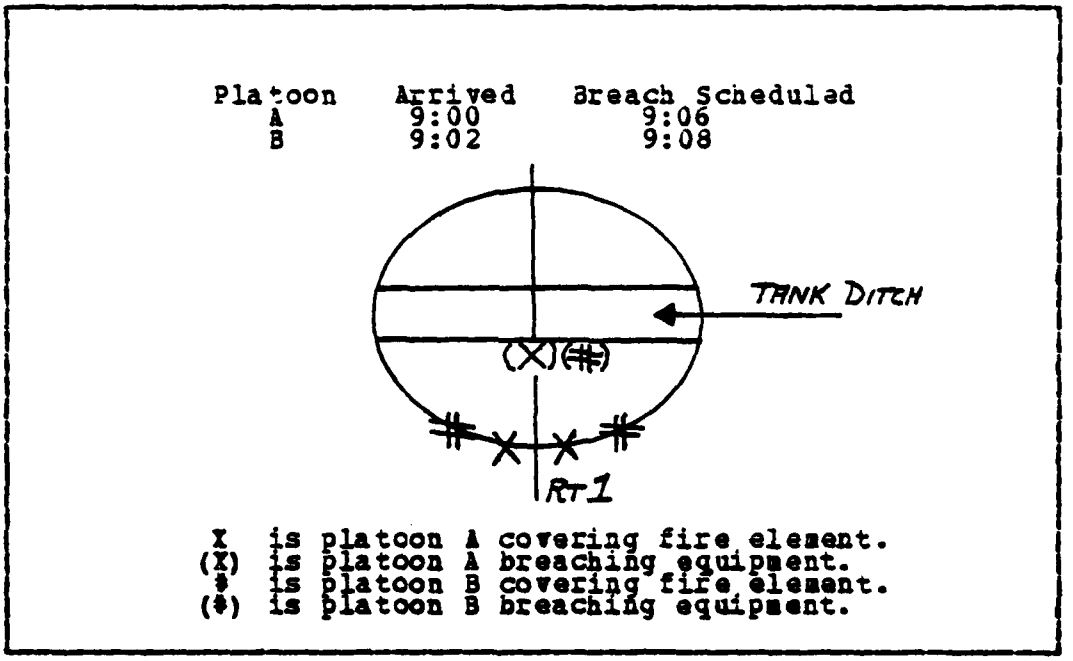


Figure 24: Multiple Breach of a Tank Ditch, Time: 9:02

This concludes the discussion of Event HEAVY.JUNK, the next explanation concerns the Event GAP.BREACH. (See Figure 25)

6. Event GAP.BREACH

Event GAP.BREACH simulates the actual break-through at the gap obstacle. At the beginning of this event, some

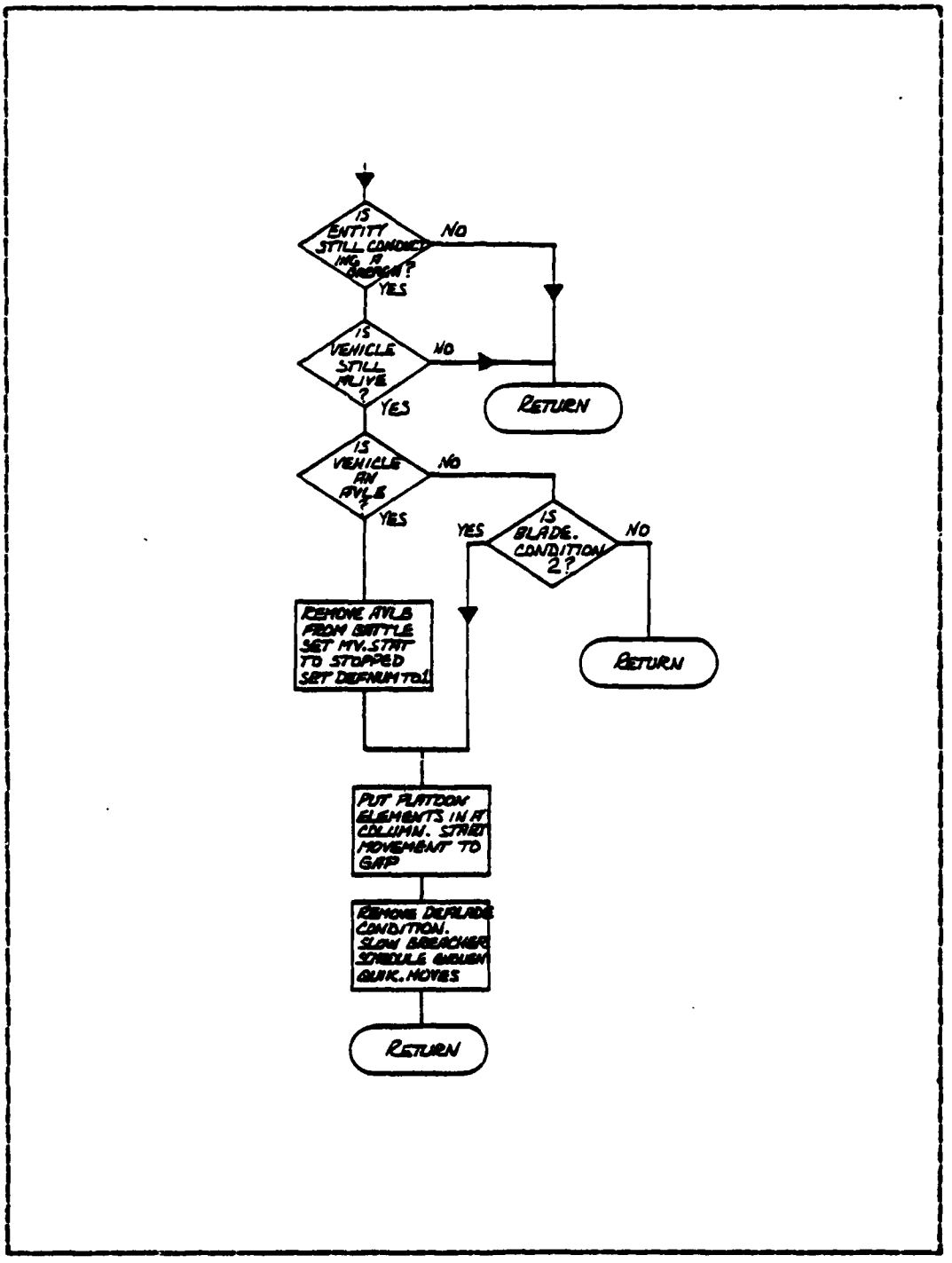


Figure 25: Event GAP.BREACH Flowchart

checks are made to ensure that the scheduling entity is still breaching and alive, and to determine the type of breaching equipment involved. If the entity is alive, breaching, and an AVLB, then it completes the bridge launching operation. At this time, the bridge launcher vehicle is put in full defilade and the AVLB as a participating entity is removed from the simulation. The assumption is made, that once a bridge is emplaced over a gap; the launcher remains at the bridge site and follow-on elements are allowed to cross over the gap on this bridge. Irregardless, of the type of equipment performing the breach, the remaining platoon elements are put into a column formation and movement is started to or from the gap depending on the element's present location. In addition, all vehicles are taken out of any defilade postures, the breaching equipment is allowed to shoot again, and is slowed to allow other (covering fire) platoon elements time to catch up. The knowledge level of the obstacle is changed to three to signify that this route has a breach. The breach in progress flag in TRAP.CONTROL is reset to zero (see Appendix A).

This concludes the discussion of the internal actions that occur at a gap obstacle. In the remaining section of this chapter, the actions taken upon exit from a gap obstacle field are discussed. (See Figure 26)

7. Gap Exit Actions

When an entity activates a gap obstacle field exit action, it is checked to determine if the entity is a vehicle. If the entity is a vehicle moving in the attack direction, it affects the knowledge level of the obstacle. Dismounted entities/soldiers are not allowed to affect the breach status of a gap obstacle. The status of the obstacle is again checked to determine if it is activated or just planned. If the entity is not a dismounted soldier and the obstacle is indeed a barrier to movement then the departing entity's speed is increased to the maximum allowed by the terrain and all vehicle attributes affected by the gap field are reinitialized. The last check determines if the entity is moving in the direction of the attack following a successful gap crossing or in the opposite direction, returning to the decision ellipse after an uneventful breaching attempt. If the latter is true, nothing further happens in the gap obstacle and the entity moves back to the

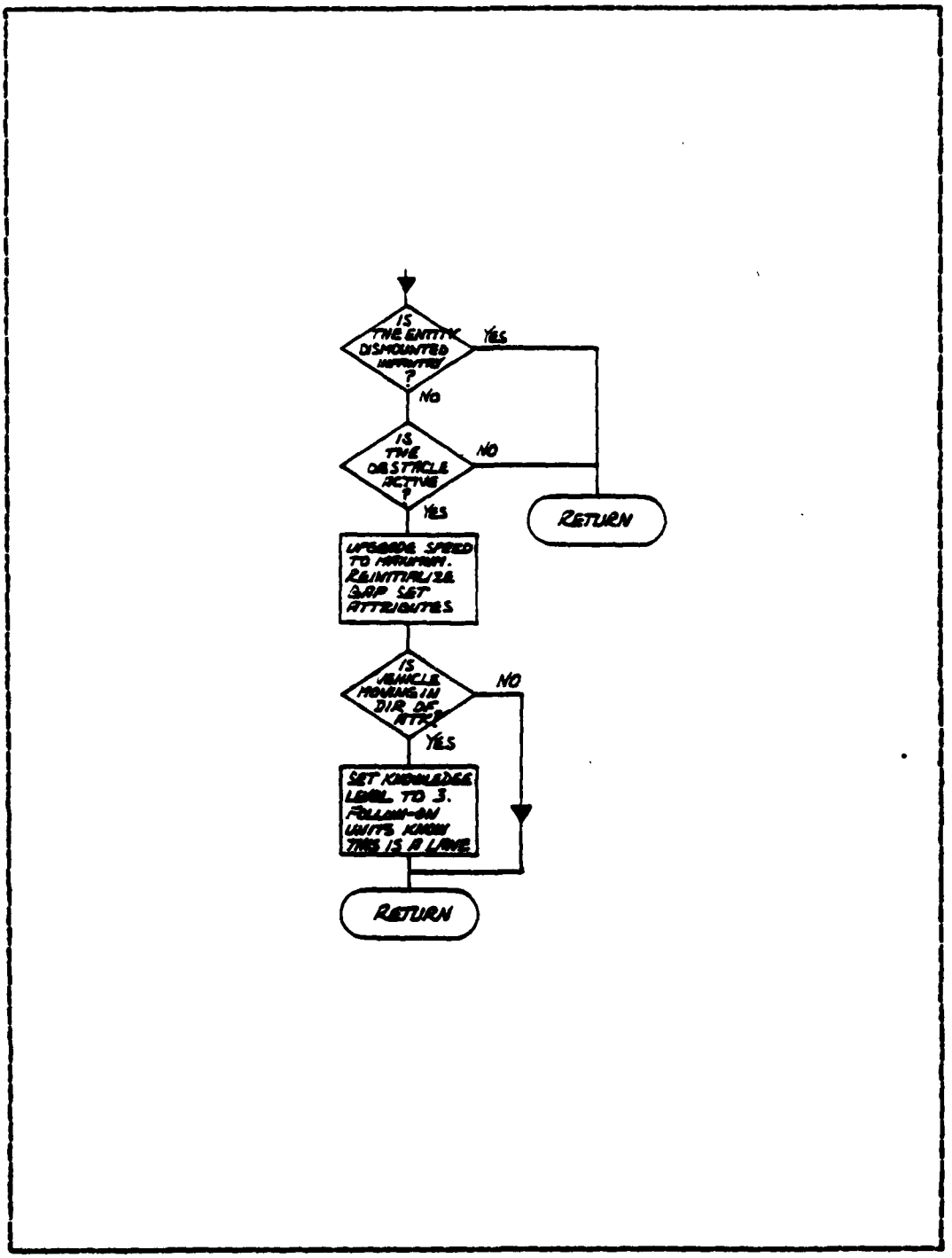


Figure 26: Routine GAP.LEAVE Flowchart

decision ellipse to select a lane or bypass the obstacle. If, however, the vehicle has passed through the crossing site and is continuing the attack, then the obstacle knowledge level is changed to three.

The synergistic effects of gap type obstacles have been described in detail. In the next and final chapter, the discussion will focus on some further enhancements to the Engineer Effects Module.

VII. FURTHER ENHANCEMENTS

The Engineer Effects Module for the STAR Combat Model has been discussed in detail. This chapter suggests some possiblities for future efforts to enhance the capabilities of this module. Many of these ideas are ones that time did not allow to be implemented in this first attempt at the module. Still, others were not thought of until the completion of the module and its associated documentation. In any event, there is still much to be done in the Engineer arena and these ideas may be helpful in future endeavors.

A. FURTHER CAPABILITIES OF THE OBSTACLES REPRESENTED

Scatterable minefields exist in the current module in both an activated or a deactivated state. This acts as an on and off switch for this obstacle. The Field Artillery could shoot scatterable mine rounds into an unactivated, preplanned, scatterable minefield ellipse and then activate this field by setting its attribute, P1.FLD, to the number of mines delivered in this field.

Most scatterable mines possess a self destruct mechanism. It is possible to simulate this characteristic

by scheduling an event to set the attribute P1.FLD of a Scatterable minefield, to zero at some time in the future of the simulation. The time duration between the activation and deactivation time of the minefield should be the self destruct time of the mines in the field.

Further, the detonation of a scatterable mine could be affected by the probability of the mine being a dud, self destructing early, or not ejecting from the shell properly. For a more detailed discussion of these probabilities and effects, see Reference 28.

B. ENGINEER UNIT REPRESENTATION IN STAR

Engineer units are not currently represented in STAR. Now that the primary obstacles and their effects are modelled, a possible future iteration in the enhancement of the Engineer Module is the explicit modelling of Engineer units.

1. Engineers in the Offense

The discussion in Chapter two avoided the deliberate breach as an option for the tactical commander to utilize to overcome obstacles. If the Engineer units are specifically represented, then this tactical option becomes a reality on the simulated battlefield. Line charges and the Surface

Launched Unit Fuel Air Explosive system (SLUFAE), both organic to Engineer units could also be simulated. [Ref. 20]

2. Engineers in the Defense

Again, if the Engineer units are specifically represented, then their employment in conjunction with planned, inactivated obstacle ellipses to simulate obstacle construction, while the battle is ongoing could be a reality. This could be accomplished by planning to send Engineer units on routes through several unexecuted obstacle ellipses. Upon arrival at an inactivated obstacle field, an event to activate the obstacle could be scheduled to occur at some time in the future. The time until activation is the emplacement/excavation time required for the particular obstacle. This time parameter can be input as a field attribute. The Engineer units would be required to monitor the disposition of friendly forward elements. This is to determine if activation of the obstacle is still possible or if an early/uneventful field departure by the Engineers will occur. In any event, the Engineer unit would move to the next field obstacle ellipse planned on its route. Engineer units could either activate obstacles themselves or turn over prepared targets to tactical units for them to

detonate/activate. This cycle of events would continue until the Engineers reach their final defensive position.

C. EQUIPMENT ENHANCEMENTS/CAPABILITIES

The effectiveness of the mine roller or mine plow for clearing mines may be of some concern. If this is the case, the number of hits could be tallied in routine POP.A.MINE using the existing framework for determining the hit location of a mine in relation to the activating vehicle. The only dimensions required are the skid width of the mine plow or the roller width of the mine roller. In addition, a roller or mine plow could be eliminated after reaching some threshold value of mine hits or each hit could be Monte Carlo'ed against a probability of kill of a roller or a plow.

The Armored Vehicle Launched Bridges (AVLB) that span a gap are currently removed from the simulation as soon as the bridge is launched. If the user wishes to reactivate this employed bridge, then an event to change the launched status of the bridge vehicle would have to be scheduled to occur at some future time. This event would alter the value of a bridge vehicle's NV.STATE attribute, change its full defilade posture, and file it in new PLATOON, COMPANY, and

BATTALION sets. In addition, the breach status on a route through an obstacle would have to be changed to reflect the removal of the bridge from the gap.

D. ADDITIONAL OBSTACLE REPRESENTATION

A built up area could be represented by an ellipse that would slow and channelize movement. Incoming and outgoing direct fires from this ellipse could be modified or halted in order to simulate vehicles being masked by buildings in the built up area. This could be accomplished by changing the entity's defilade posture to halt incoming direct fire and either its plow or blade conditions to halt outgoing direct fire. Values of four or two, for PLOW.COND or BLADE.COND, respectively, would accomplish this effect.

A river crossing site ellipse could be used to simulate vehicles crossing a river by swimming or snorkeling. This could be the "bull through" option, near a blown bridge obstacle. Time could be assessed against tanks that had to affix snorkeling fixtures, their speed could be degraded, direct fire stopped, and they could be put in a full defilade posture (underwater). The amphibious vehicles would have similar actions taken to degrade their performance.

E. OTHER TACTICAL OPTIONS

The option to cover a deployed mine breaching vehicle by platoon or company direct fire should be added to this Engineer Module. The basic logic and structure for its implementation already exists in the gap obstacle portion of the module.

Imperfect or nonexistent intelligence between attacking echelons could be simulated by zeroing out the TRAF.CONTROL matrix and re-initializing the knowledge level of the obstacles, field attribute P6.FLD, between passage of the echelons.

The Engineer Effects Module currently simulates the following tactics.

1. Breach the obstacle if possible, otherwise "bull through".
2. Breach the obstacle if possible, otherwise select a clear lane if one exists, or bypass the obstacle.

Additional tactical options could be modelled with minor modifications to existing code. This may require the addition of another attribute to the obstacle ellipse to indicate the appropriate tactical option. Some additional tactical options include the following:

1. Select a clear lane if one exists, otherwise bypass, breach or "bull through" as a last resort.
2. Breach the obstacle if possible, otherwise select a clear lane if one exists, or "bull through" as a last resort.

7. MISCELLANEOUS

In the current Engineer Module, AVLB vehicles are prevented from firing by giving them no ammunition, however, they still search for targets. In the new Night Vision Lab Search Model, target selection by AVLB's can be turned off by defining a search type which calls for a search tactic of zero. A search tactic of zero does not exist and the simulation will only call one search for the bridges. However, the flash detection may be an exception to this rule. Thus, a search or detect will never occur for a bridge entity again. This would yield a small savings in computer time, as a result of fewer searches during the model run.

In this chapter, the discussion has centered around ideas to further enhance the capabilities of the Engineer Effects Module. In a modelling project of this magnitude, there is no fitting conclusion. As each piece of the Combat Modelling process is completed, more questions pertaining to the whole process are generated. Perhaps this is what makes Combat Modelling the iterative process it is. One statement seems to suffice...The effort goes on.

APPENDIX A

EXPLANATION OF THE ARRAY TRAF.CONTROL

In this appendix, the dimensioning and function of the array TRAF.CONTROL is discussed. This array is the key to the implementation of pseudo-dynamic movement in the Engineer Effects Module for STAR. The array TRAF.CONTROL is accessed many times throughout the simulation to obtain information necessary for pseudo-dynamic movement. In order to provide the reader with a more complete picture of TRAF.CONTROL, it is described in detail in this appendix.

A. THE DIMENSIONING OF TRAF.CONTROL

TRAF.CONTROL is a three dimensional real array with the following subscripts:

OB: obstacle number (1,2,...OB.NUM)

RT: route number (1,2,...RT.OB.NUM)

K: integer index (1,2,3,4,5)

The number of obstacles (input parameter - OB.NUM) and the number of affected routes (input parameter - RT.OB.NUM) each play a critical role in the dimensioning of this array. There are several ways to obtain these input parameters,

some save computer time by demanding some input manipulation, while others use more computer time but are more user friendly. The more efficient example is described first.

1. If the user puts the obstacle ellipses first on the input list, then OB.NUM will be the smallest possible value. Since the first dimension of TRAF.CONTROL is based on OB.NUM this action makes the first dimension of TRAF.CONTROL the smallest it can be.
2. For the second dimension of TRAF.CONTROL, if the user takes the routes that are affected by the obstacles and puts them first on the input list, the RT.OB.NUM will be the smallest possible value. Since the second dimension of TRAF.CONTROL is based on RT.OB.NUM, this step makes the second dimension of TRAF.CONTROL also the smallest possible value.
3. The third dimension of TRAF.CONTROL is indexed from 1 to 5.

Since the above procedure provides TRAF.CONTROL with the smallest possible dimensions, it is the most efficient method of dimensioning TRAF.CONTROL. It is true that the SIMSCRIPT language possesses the capability to utilize

ragged arrays. The authors pondered over this very issue and made a conscious decision not to use this capability. Since TRAF.CONTROL is accessed so often in the Engineer Effects Module, the authors determined that direct access to its contents was more desirable than the numerous searches of array data required by the ragged array technique. Other alternatives to the manipulated route and obstacle approach listed above range from using just step number one above and not ordering the routes to using step number two and putting the fields in any order or finally, not ordering anything, thus creating a TRAF.CONTROL array that is very large and inefficient in terms of computer storage. In this last case, OB.NUM is the total number of fields used in the simulation and RT.OB.NUM is the highest route number used.

With the dimensioning of TRAF.CONTROL explained, the remaining discussion will focus on the contents of TRAF.CONTROL and its function in the Engineer Effects Module.

B. THE CONTENTS OF TRAF.CONTROL

TRAF.CONTROL contains information that is slightly different for the two major types of obstacles modeled. This section discusses the values with the obstacle

differences taken into consideration. The array TRAF.CONTROL is accessed by obstacle number, route number, and the third dimension value. The third dimension values in each of the five positions are the values that are required by the simulation. It is for this reason that the reader is reminded that the values being described in the five positions exist for each obstacle and associated route through the obstacle. The values that are stored in these five places are listed by major obstacle type below:

1. Minefield TRAF.CONTROL Contents

These are the items contained in each of the five positions of the array TRAF.CONTROL with regard to minefields.

1 The route number, the entering entity is travelling on in the minefield. This number is accessed by follow-on units when this is a cleared lane through the minefield.

2 The Movement Control Point in the decision ellipse or a value of zero, if the entering entity is going to "bull through" the minefield.

3 The cumulative distance clear in the minefield on this route. If this value is plus infinity, this route is a lane through the minefield.

4 The most recent distance to mine encounter for the lead vehicle in the lead platoon trying to traverse the minefield.

5 The total number of mine encounters scheduled thus far for the elements travelling on this route through the minefield. (bump number)

2. Gap Obstacle TRAF.CONTROL Contents

These are the items contained in each of the five positions of the array TRAF.CONTROL with regard to gap obstacles.

1 The route number that the activating entity is travelling upon entry of the gap obstacle ellipse. This number is accessed by follow on units when this route has a breach/span through the obstacle.

2 The Movement Control Point in the decision ellipse or a value of zero, if the entering entity is going to "bull through".

3 The lane status of this obstacle for a particular route.

0 - a lane does not exist on this route.

+ infinity - a lane exists on this route through the obstacle.

4 The breaching status for this route. If the value is 0, then there is no breach in progress on the route. Any other number indicates the platocn number of the platoon currently attempting a breach on the route.

5 not used.

C. THE FUNCTION OF TRAF.CONTROL

The values stored in this array differ for the major obstacle types, thus, the discussion is categorized in this manner.

1. TRAF.CONTROL Function in Minefields

The first position in TRAF.CONTROL contains the route number of the entity travelling in the minefield. This route number is entered in TRAF.CONTROL upon entry into a minefield by an entity. The second position pertains to the tactical option to "bull through" or to bypass the minefield. The options to "bull through" or to be able to bypass, are values specified as attributes by the user. These values determine what value will be placed into TRAF.CONTROL position number two. If the user specifies that the "bull through" option is to be used, i.e. there is no bypass route, then the entity conveys a value of zero to

this postion upon entry into a decision ellipse. If the entity has the option to bypass the minefield, then the entity conveys the number of the Movement Control Point in the decision ellipse to position number two. At this point the simulation has all the necessary information to perform pseudo-dynamic route selection of cleared lanes in the minefield. If a clear lane through the minefield exists, and entities have decision ellipses and the bypass option, then the TRAF.CONTROL array is searched over position three to locate a lane. The affected units are then redirected to the clear route and MCP, stored in array TRAF.CONTROL for the obstacle.

The remainder of the discussion of TRAF.CONTROL pertaining to minefields concerns positions three, four , and five. Each vehicle (entity) has an attribute called the tank bump number (TBUMP.NUM). This attribute allows the entity to move as a member of a platoon when in a minefield. Upon entry into a minefield the value of TBUMP.NUM is always zero. If the bump number (position number five) in TRAF.CONTROL is greater than the TBUMP.NUM of the vehicle entering the field, then another vehicle has preceded it into the field. Since another vehicle is already some

cleared distance in the minefield, the entering vehicle is allowed to travel that distance with no mine effects. In order to implement this effect, TRAF.CONTROL is utilized further.

The difference between the TBUMP.NUM and the bump number stored in TRAF.CONTROL is computed. If the difference is one, the entering vehicle receives the most recent distance to a mine encounter for the platoon lead vehicle. This value is stored in position four of TRAF.CONTROL. If the difference is more than one, the vehicle's next mine encounter distance is set to the cumulative distance clear for this route and stored in the TRAF.CONTROL matrix, position three. The vehicle's TBUMP.NUM is updated to the value of the TRAF.CONTROL bump number. If, however, the difference between TBUMP.NUM and the bump number in position five is zero, then the vehicle is the first from the platoon to enter the minefield. Routine MINE.SCHEDULE is called and a distance to a mine encounter is computed. This distance value is placed in TRAF.CONTROL positions three and four and both the array bump number (position five) and the vehicle's TBUMP.NUM are set to one.

The use of the bump numbers at the minefield internal actions is nearly the same as in the discussion above. The difference between the vehicles TBUMP.NUM and the value of the TRAF.CONTROL array bump number is calculated. If the difference is zero, then this entity is the lead element in the platoon. The next mine encounter distance is obtained from routine MINE.SCHEDULE and this distance is placed in position four of TRAF.CONTROL. This distance is then added to the valre (cumulative distance clear on the route) already stored in TRAF.CONTROL position three. The entity's TBUMP.NUM and the array bump number are both incremented by one. If on the other hand, the TBUMP.NUM of the activating entity is less than the bump number (position five) of the TRAF.CONTROL matrix, then other actions take place. If the difference is only one, then the distance stored in position four of TRAF.CONTROL will be transmitted to the activating entity as his next mine encounter distance and the entity's TBUMP.NUM is increased by one. If the difference is more than one mine encounter distance, then the distance from the activating entity to the lead platoon vehicle's projected cleared location is calculated and this becomes the distance to the

next mine encounter for the activating entity. This entity's TBUMP.NUM is given the value of the array's bump number (position five). This process continues until the platoon traverses this minefield or is depleted. When an entity exits a minefield after a successful breach, then the cumulative distance clear value for the route, TRAF.CONTROL storage location three, is set to plus infinity. This signifies that a breach exists through the minefield on this route and allows follow-on units to search TRAF.CONTROL to find the cleared lane for this obstacle.

2. TRAF.CONTROL Functions In Gap Obstacles

The same actions for the TRAF.CONTROL array for positions number one and two occur for the gap type obstacles. The storage for positions three, four and five differ from the previous discussion concerning minefield interactions. Position number five is not used by entities involved in gap obstacles. The third storage location is used as a flag to trigger dynamic lane selection. If it contains a value of plus infinity, then a successful breach has been performed through this tank ditch and follow-on unit movement is redirected toward this lane. If the value contained in position three is zero, then the gap has not

been penetrated and units must either attempt a self breach, breach with equipment, or bypass this excavated obstacle. TRAF.CONTROL position number four contains the number of the platoon that is already performing a breaching operation on this route at the gap. This information gives follow-on platoons, that would otherwise have to attempt a self breach, the opportunity to cover the breaching platoon's effort by fire. In addition, if the breach attempt is successful, the follow-on platoon can utilize the breach gained for their crossing of the gap.

The discussion of the TRAF.CONTROL array is, indeed an important one because this array organizes the movement of platoons and enables units to dynamically select cleared lanes. In addition, TRAF.CONTROL coordinates fire support of current breaching efforts from follow-on units on the same route in the instance of a gap obstacle. This concludes the explanation of the array TRAF.CONTROL.

APPENDIX B

MODIFIED ROUTINES AND EVENTS FROM BASIC STAR

In the construction of this model, many changes were made to the existing STAR code. In order to facilitate the location of these changes the authors have supplied several lines of unaffected code occurring before and after the changed lines. In addition, the lines with changes are further highlighted through the use of a plus sign (+) immediately following the line number.

1. PREAMBLE

Purpose

The PREAMBLE provides the compiler with definitions regarding entities, attibutes, and sets; events and routines; background mode, type and dimensionality and global variables and arrays.

Modifications

The following additions have been made to the PREAMBLE. For additional information refer to the thesis section listed below.

Events (Appendix C)

DITCH.KILL
DIVERT
GAP.BREACH
GAP.JOCK
HALT
HEAVY.JUNK
QUIK.MOVE
STAND.TO
TURN.AROUND
WALL.BREACH

Global Variable

FLD.POINTER (1-D) Integer

This 1-dimensional array contains the temporary field pointers. This enables the user to access the obstacles by using the sequential order numbers given the fields on input.

TRAF.CONTROL (3-D) Real (Appendix A)

Permanent Attributes Integer

GRUNTLETH (3-D)

This 3-Dimensional array contains the BRL casualty data for anti-personnel mines.

MINLETH (4-D)

This 4-Dimensional array contains the BRL casualty data for anti-tank mines.

PLT.COND (1-D)

This attibute of the platoon leader keeps track of the present action status of the platoon in a minefield (i.e. plowing, offsetting, pushing, etc.)

NXT.BELT (1-D)

This platoon leader attribute stores the number of the next mine belt to be encountered in the minefield. It allows the platoon to move through a minefield as an organized unit.

Temporary Attribute Integer

BLADE.COND

This unit attribute gives the tank the ability to have a dozer blade characteristic.

FLD.FORM

This unit attribute gives the entity the ability to change formations when encountering an obstacle.

HOLD.FORM

This unit attribute is a storage place, where the units formation number for the platoon (FORMACODE) can be placed when lateral route movement is appropriate.

PLOW.COND

This unit attribute gives the tank the ability to have a mine plow characteristic.

TBUMP.NUM

This unit attribute allows the entity to move in a very detailed manner as a member of his platoon when in an obstacle field.

TPNAM.FLD

This field attribute keeps the temporary pointer number of the field.

WHAT.BELT

This unit attribute works in conjunction with the platcon leader attribute (NXT.BELT).

In order for an event to have arguments passed from event to event or routine to event, etc., the arguments must to be given in the PREAMBLE.

The following arguments will be described in Appendix C.

BAGGED.BOY
BREACHER
BYPASS
GULLY.CAT
HOLE
HOME
INT.TIME
NUM.POINTER
RAMMER
REP.NUMBER
SABOT.SHOOTER
SIDE.STEPPER
SLOW.DOWN
STATUS
TEAR.DOWN
TWIST

Temporary Attribute Real

ANGLE.FLD

This field attribute carries the orientation angle in radians measured counterclockwise from east to the major axis.

AREA.FLD

This field attribute carries the area of the field in meters squared.

FSPEED.FAC

This unit attribute gives the entity the ability to slow down because of obstacle encounter.

P5.FLD (Appendix D)

P6.FLD (Appendix D)

P7.FLD (Appendix D)

P8.FLD (Appendix D)

P9.FLD (Appendix D)

P10.FLD (Appendix D)

P11.FLD (Appendix D)

P12.FLD (Appendix D)

SAMAJ.FLD

This field attribute is the semi-major axis length of the elliptical field in meters.

SAMIN.FLD

This field attribute is the semi-minor axis length of the elliptical field in meters.

Brief Explanation

The additions are shown as they appear in the code. Enough of the basic STAR code is given so that the additions can be quickly added. As mentioned previously, a + sign immediately after the line number highlights the lines that have been changed. This procedure will be used throughout the remainder of this appendix to annotate changed basic STAR code.

CODE

```
 1  PREAMBLE
 2
12  THE SYSTEM HAS A  LE6D(*/4), A LE5D(*/4), A DLA(*/4),
13       A MVARY(*/4), A ARRAY(*/4), A PHSCORD(*/4),
             A BN.AIR.PRI(*/4),
14+      A BNCORD(*/4), A COCORD(*/4), A RTAB(*/4),
             A MINLETH(*/4),
15+      A GRUNTLETH(*/4), A LE12(*/4), A LE72(*/4),
             A LE31(*/4),
16       A LE61(*/4), A LE81(*/4), A LE83(*/4), A LE11(*/4),
             AND A LE71(*/4)
30+ DEFINE BNCORD, COCORD, TABLE, RTAB, USMG, GRUNTLETH,
             DATA.SP,POINT.HOLD
31+  AS INTEGER 3-DIM ARRAYS
32+ DEFINE  SOVMG, MINLETH AS  INTEGER 4-DIMENSIONAL ARRAYS
101  ''
102  ''
103  ''   SEVERAL OWNS FOLLOW
104  ''
105  EVERY SQUAD.LEADER OWNS A SQD.UNIT
106  EVERY PLATOON.LEADER HAS A (R4HAT(1-15),R3HAT(16-32)),
             A   (R2HAT(1-15),
107+      R1HAT(16-32)), A (ROHAT(1-30),GONE(31-32)),
             A NXT.BELT, A PLT.COND
108       AND OWNS A PLT.UNIT
109  EVERY COMPANY.COMMANDER HAS A P.TO.AREAS,
             A   (COWT(1-5),CREQST(6-7),
110       COMPY(8-32)),  OWNS A COMP.UNIT  AND  MAY BELONG
             TO A BATTALION
111  EVERY BN.COMMANDER HAS A (BNWT(1-5),BNCUR(6-11),
             BNLO(12-17),BNGO(18-23),
112       BATT(24-30),BREQST(31-32)),  OWNS A BATTALION AND
             MAY BELONG TO A BRIGADE
113  EVERY BDE.COMMANDER HAS A  (BDECUR(1-10),BDELO(11-20),
             BDEGO(21-32))
114       OWNS A BRIGADE, AND MAY BELONG TO A DIVISION
115  ''
116  DEFINE S1DED,S2DED,S3DED,S4DED,S5DED,S6DED,MV.UNIT,
```

```
            CUDED,S1SRT,S2SRT,
117+        S3SRT,S4SRT,S5SRT,S6SRT,CUSRT,PRTY,REDSIN,REDRG,
            BLUSIN,PLT.COND,
118         R1DED,R2DED,R3DED,R4DED,R5DED,RV.UNIT, R6DED,
            RUDED,R1SRT,R2SRT,
119         RETY,R3SRT,R4SRT,R5SRT,R6SRT, RUSRT,XCENT,YCENT,
            R4HAT,R3HAT,
120+        R2HAT,R1HAT,R0HAT,GONE,NXT.BELT, P.TO.AREAS,COWT,
            CREQST,COMPY, BNWT,
121         BNCUR,BNLO,BNGO,BATT,BREQST, BDECUR,BDELO,BDEGO
            AS INTEGER VARIABLES
122     EVERY DIV.COMMANDER HAS A DIVWT, A DIVCUR, A DIVLO,
123       A DIVGO, A DDIV, A DIVREQST, A NO.DIV.UNIT,
124       AND OWNS A DIVISION
125     ''
214     ''
215     TEMPORARY ENTITIES
216     ''
217     GENERATE LIST ROUTINES
218     EVERY FIELD HAS A NAM.FLD, A XC.FLD, A YC.FLD,
            A PXX.FLD, A PYY.FLD,
219+      A PXY.FLD, A TYP.FLD, A ANGLE.FLD, A SAMAJ.FLD,
            A SAMIN.FLD,
220+      A AREA.FLD, A TPNAM.FLD, A P1.FLD, A P2.FLD,
            A P3.FLD, A P4.FLD,
221+      A P5.FLD, A P6.FLD, A P7.FLD, A P8.FLD, A P9.FLD,
            A P10.FLD,
222+      A P11.FLD, A P12.FLD AND MAY BELONG TO A FLD.SET
223+   DEFINE XC.FLD,YC.FLD,PXX.FLD,PYY.FLD,PXY.FLD,
            ANGLE.FLD,SAMAJ.FLD,
224+   SAMIN.FLD,AREA.FLD,P1.FLD,P2.FLD,P3.FLD,P4.FLD,P5.FLD,
            P6.FLD,P7.FLD,
225+   P8.FLD,P9.FLD,P10.FLD,P11.FLD,P12.FLD AS REAL
            VARIABLES
226+   DEFINE TYP.FLD, NAM.FLD,TPNAM.FLD AS INTEGER VARIABLES
227     EVERY UNIT HAS A (NAME(1-11),COLOR(12-12),
            SYS.TYPE(13-16), WPN.TYPE(17-23),
228         VEH.TYPE(24-27),ALIVE.DEAD(28-29),DEFNUM(30-32)),
            A
229         (KHZ.LAST.TRGT(1-24),C.2(25-30)),  A  ROUND,
            A   (ND.HIT(1-4),
230         M.HIT(5-7),F.HIT(8-10),MF.HIT(11-14),K.HIT(15-17),
            NUM.HIT(18-22),
231         FIRED.AT(23-27),MKILL(28-28),FKILL(29-29),
            MFKILL(30-30), KKILL(31-31)),
232         A (SECLDR(1-24), SQDVEH(1-24), CBAR(25-32)),
            A T.SPD, A MICRO, A HIT.STATE,
233         A  (SEC(1-10),PLT(11-18),CO(19-23),BN(24-28),
            FIP(29-30), SCHED(31-32)),
234         A  (RGT(1-5),PI.HAT(6-8),MINE.DET(9-11),
            MV.STATE(12-14),CSS(15-15),
235         M.BLUE.ALIVE(16-16),M.RED.ALIVE(17-17),
            M.COMP.UNIT(18-18),
236         M.PLT.UNIT(19-19),M.TANKS(20-20),HITSHOT(21-23),
            MISSSHOT(24-27),
237         PROJO(28-32)),  A  (COCDR(1-24),PH(25-27),
            CRF(28-32)),
238         A  (PLTLDR(1-24),TRF(25-32)),  A  (CARR(1-4),
            R.CON(5-7), NEXT.HCP(8-14),
239         CP.NEXT(8-14), POS.POINT(15-19),ROUTE(20-28),
            I.ROUTE(20-28),
240         FORMACODE(29-32)), A SPD, A  (RANGE(1-16),
            AREA(17-25),VAREA(17-25),
241         AP.TOW(26-32),AMMO1(26-32)),  A  (HE.DRAG(1-8),
            AMMO2(1-8),
242         AW1.OR.MSL3(9-20),AMMO3(9-20), AW2.OR.ADM(21-32),
            AMMO4(21-32)),  A
243         DIR.OF.MVMT,  A PRI.DIR,  A LST.DIR,  A X.CURRENT,
```

```
          A Y.CURRENT,
244       A Z.CURRENT,  A M.D,  A F.D,  A MF.D,  A TIM.FIRE,
            A (FOE(1-24),
245       C.1(25-30),TSKED(31-31)),A  (POS.IN.PLT.AREA(1-6),
            FORMPOS(7-12),
246       START.AREA(13-22),END.AREA(23-32)),A (FOM(1-1),
            DIRC.ON.RT(2-2),
247       R.D.STATUS(3-4),SCH.TO.MOVE(5-6),GRNDATK(7-8),
            FIR.MODE(9-9),
248       N.HD(15-19),N.AB(20-25),N.TH(26-32)), A FLD.NO,
249+      A (WHAT.BELT(1-10),FLD.AKT(11-16),N.PLE(17-25),
            N.ARM(26-32)),
250+      A (FA(1-8), ARTY.HITS(9-18), FLD.FORM(19-25),
            TBUMP.NUM(26-32)),
251+      A (PLOW.COND(1-8), HOLD.FORM(9-19),
            BLADE.COND(20-32)),
252+      A FSPEED.FAC, A P.BODY.INCAP,A P.HD.INCAP,
253       A P.TH.INCAP, A P.ARM.INCAP, A P.AB.INCAP,
            A P.PLE.INCAP,
254       A FLD.INT.DIST, A FLD.BDY.DIST,
255       A POP.UP.TIME,  '' USED FOR COMING OUT OF
            FOXHOLE AFTER FA IMPACT
256     AN FMTANK.LIST,   '' FOR COMMO
257   A (PROLE(1-2),MROLE(3-5),NO.OBS(6-8),RDRON(9-9),
            THERMAL(10-10), RWR(11-11),
258   RADAR(12-12),POP.UP(13-13),BASE(14-20),LSS(21-21),
            LSD(22-22),  MODE(23-25),
259   LSRON(26-26),CG.MUNITION(27-32)),A (CURR.SORTIE(1-24),
            CP.FINAL(25-32)),
260   AN (ORB.RT(1-24),PHSE(25-30),RDR.PAINT(31-32)),
            A ZZANGLE,
261   A (BURST.SIZE(1-4),N.BURSTS(5-8),RELOAD.CT(9-19),
            DET.SEC(20-20), DET.PLT(21-21),
262   SEL.SEC(22-22),SEL.PLT(23-23),MSN.ABORT(24-24),
            LWR(25-25)), A PBH,
263   A GUIDE.OFF, AN EXP.TIME, A (PASSOFF(1-24),
            NO.FIRE(25-25))
264       AND  MAY BELONG TO A TANKS, A BLUE.ALIVE,
            A RED.ALIVE, A COMP.UNIT,
265       A PLT.UNIT AND A SQD.UNIT
266   ''
267      INHIBIT LIST ROUTINES
268   DEFINE NAME,COLOR,SYS.TYPE,WPN.TYPE,VEH.TYPE,
            ALIVE.DEAD,DEFNUM,
269       KHZ.LAST.TRGT,C.2, ROUND, ND.HIT,M.HIT,F.HIT,
            MF.HIT,K.HIT,NUM.HIT,
270       FIRED.AT,MKILL,FKILL,MFKILL,KKILL, SECLDR,CBAR,
            SEC,PLT,CO,BN,PIP,
271       SCHED,RGT,PI.HAT,MINE.DET,MV.STATE,CSS,
            M.BLUE.ALIVE,M.RED.ALIVE,
272+      M.COMP.UNIT,M.PLT.UNIT,M.TANKS,HITSHOT,MISSSHOT,
            PROJO,PLOW.COND,
273+      COCDR,PH,CRF, PLTLDR,TRF, CARR,R.CON,NEXT.MCP,
            CP.NEXT,WHAT.BELT,
274       POS.POINT,ROUTE,I.ROUTE,FORMACODE, RANGE,AREA,
            VAREA,AP.TOW,AMMO1,
275       HE.DRAG,AMMO2,AW1.OR.HSL3,AMMO3,AW2.OR.ADM,AMMO4,
            FOE,C.1,TSKED,
276       POS.IN.PLT.AREA,FORMPOS,START.AREA,END.AREA,FOM,
            DIRC.ON.RT,
277+      FLD.FORM,TBUMP.NUM,BLADE.COND,HOLD.FORM AS INTEGER
            VARIABLES
278   ''
279   ''INFANTRY INTEGER VARIABLES FOR UNITS
280   DEFINE FLD.NO,FLD.AKT,N.ARM,N.TH,N.HD,N.AB,N.PLE,
            FIR.MODE,GRNDATK,
281     SCH.TO.MOVE,SQDVEH AND R.D.STATUS AS INTEGER
            VARIABLES
```

```
282   ''
283   ''INFANTRY REAL VARIABLES
284   DEFINE FLD.BDY.DIST,FLD.INT.DIST,P.BODY.INCAP,
            P.HD.INCAP,P.TH.INCAP,
285     P.AB.INCAP,P.ARM.INCAP,P.PLE.INCAP,H.IN,TH.IN,
            AB.IN,PLE.IN,ARM.IN,
286+    FSPEED.FAC AND TIM.IN AS REAL VARIABLES
462   ''
463+  ''   ENGINEER EVENTS
464+  ''
465+  EVERY QUIK.MOVE HAS A NUM.POINTER, A REP.NUMBER, AND
            A INT.TIME
466+  EVERY STAND.TO HAS A BAGGED.BOY
467+  EVERY HALT HAS A SLOW.DOWN
468+  EVERY TURN.AROUND HAS A TWIST
469+  EVERY DIVERT HAS A SIDE.STEPPER, A BYPASS, AND A HOME
470+  EVERY WALL.BREACH HAS A RAMMER, AND A TEAR.DOWN
471+  EVERY DITCH.KILL HAS A GULLY.CAT
472+  EVERY HEAVY.JUNK HAS A STATUS
473+  EVERY GAP.BREACH HAS A BREACHER, AND A HOLE
474+  EVERY GAP.JOCK HAS A SABOT.SHOOTER
475+  DEFINE NUM.POINTER, REP.NUMBER, BAGGED.BOY, SLOW.DOWN,
476+   TWIST, SIDE.STEPPER, BYPASS, HOME, RAMMER, GULLY.CAT,
            STATUS,
477+   BREACHER, HOLE, TEAR.DOWN, SABOT.SHOOTER AS INTEGER
            VARIABLES
478+  DEFINE INT.TIME AS A REAL VARIABLE
479   ''
744   ''
745+  '' ENGINEER ARRAYS
746   ''
747+  DEFINE FLD.POINTER AS A 1-DIMENSIONAL, INTEGER ARRAY
748+  DEFINE TRAP.CONTROL AS A 3-DIMENSIONAL, REAL ARRAY
749   ''
835   END  ''OF THE PREAMBLE''
```

2. Routine BL.CREATE

Purpose

Routine BL.CREATE is used to create temporary tank entities at designated times throughout the execution of the program. The routine is called during the execution of Event BLU.FORCES.

Modifications

The following four attributes must be assigned values from the input data or as a result of the routine.

Temporary Attributes Integer

BLADE.COND
HOLD.FORM
PLOW.COND

Temporary Attribute Real

FSPEED.FAC

Brief Explanation

Line 15 PLOW.COND and BLADE.COND are assigned the values of 0-no or 1-attached.

Line 18 FSPEED.FAC is initialized to 1.0. No effect unless in field.

Line 19 HOLD.FORM is initialized to 999. This value changes only when the entity is using lateral movement. When movement is terminated, this 999 value is set back on HOLD.FORM for the entity. It serves as a flag to determine appropriate actions. To change this initial value has unpredictable consequences.

CODE

```
  1  ROUTINE BL.CREATE(A,B,MV)
 11   CREATE A TANK
 12   LET PI.HAT(TANK)=1
 13   READ NAME(TANK),SYS.TYPE(TANK),WPN.TYPE(TANK),
                SEC(TANK),PLT(TANK),CO(TANK),
 14   BN(TANK),SQDVEH(TANK),PLTLDR(TANK),COCDR(TANK),
                START.AREA(TANK),
15+ ALIVE.DEAD(TANK),FIR.MODE(TANK),PLOW.COND(TANK),
                BLADE.COND(TANK)
 16   CALL FLD.DIST(TANK)
 17   LET FLD.INT.DIST(TANK)=RINF.C
18+ LET FSPEED.FAC(TANK)=1.0
19+ LET HOLD.FORM(TANK)=999
 20   IF CO(TANK) GT COS  LET COS=CO(TANK) ALWAYS
 85   RETURN      END
```

3. Routine MOVE

Purpose

Routine MOVE is called from routine LOC to update the location of any entity to the current time of the simulation.

Modifications

The following entity attribute has been added to the MOVE routine logic. This addition now gives the user the ability to change formations as a result of encountering an obstacle field. The capability to offset around a dead vehicle is now a reality.

Temporary Attribute Integer

FLD.FORM

Brief Explanation

Lines 118, 160-165, and 195-200

Allow the FLD.FORM attribute the ability to override the normal formation procedure.

CODE

```
  1   ROUTINE TO MOVE GIVEN VEH
 78   'NEW.MCP'
 82    LET MCP = NEXT.MCP(VEH)      LET NM = MCP * 3
 83   IF MCP EQUALS 0   ''MOVE TO POSITION IN END.AREA
 84       IF POS.IN.PLT.AREA(VEH) EQUALS 0, CALL
          BEST.POS(VEH)
 85       ALWAYS
 86       LET I = PLT(VEH)       LET K
                = POS.IN.PLT.AREA(VEH) * 3
 87       FOR J = 1 TO DIM.F(POSITION(I,*,*)) WITH
                POSITION(I,J,1) EQUALS
 88             END.AREA(VEH)
 89       DO
 90             LET X.DEST = POSITION(I,J,K-1)
 91             LET Y.DEST = POSITION(I,J,K)
 92             LET DIR = POSITION(I,J,K+1)
 96       LOOP
 97       LET D.TO.MCP = SQRT.F((X.DEST
              X.CURRENT(VEH))**2 +
 98            (Y.DEST-Y.CURRENT(VEH))**2)
102       IF D.TO.MCP LESS THAN ZERO.LEVEL,
103             LET MV.STATE(VEH) = 4
104             LET DIR.OF.MVMT(VEH) = DIR
105             LET PRI.DIR(VEH)  = DIR
106             LET SPD(VEH) = 0.
107             LET FINAL = 1
111             GO TO NEW.INCR
112       ELSE
116       GO TO DIRN.COMP
117   ELSE
118+  IF FORMACODE(VEH) = 0 AND FLD.FORM(VEH) = 0
          ''GO DIRECTLY TO NEXT MCP
122     LET X.DEST = RTE.DATA(RT,NM-2)
123     LET Y.DEST = RTE.DATA(RT,NM-1)
124     LET D.TO.MCP = SQRT.F((X.DEST-X.CURRENT(VEH))**2 +
125          (Y.DEST-Y.CURRENT(VEH))**2)
130       GO TO DIRN.COMP
131   ELSE      ''MOVE ALONG ROUTE OFFSET BY FORMATION
146       IF MCP EQUALS K/3 AND D.ON.RT EQUALS 1
147             LET NX = RTE.DATA(RT,K-5)
148             LET NY = RTE.DATA(RT,K-4)
149             LET LX = RTE.DATA(RT,K-2)
150             LET LY = RTE.DATA(RT,K-1)
151             LET I = RTE.DATA(RT,K-3)
152       ELSE
156       GO TO INTERMED
157       ALWAYS
158   ALWAYS
159       LET NLX = NX-LX       LET NLY = NY-LY
160+      IF FLD.FORM(VEH) NE 0
161+         LET I = FLD.FORM(VEH)
```

```
162+       ELSE
163+          IF I EQUALS 0,  LET I = FORMACODE(VEH)
164+          ALWAYS
165+       ALWAYS
166        LET J =     FORMPOS(VEH) * 2
167        LET X.OFF = FORM.OFFSET(I,J-1)
168        LET Y.OFF = FORM.OFFSET(I,J)
169        LET THETA = ARCTAN.F(NLY,NLX)
170        LET CTH = COS.F(THETA)
171        LET STH = SIN.F(THETA)
189        LET NLX = NX - LX              LET NLY = NY - LY
190        LET ALPH =-((CX-NX)*NLX + (CY-NY)*NLY) /
                (NLX*NLX + NLY*NLY)
191        LET PX = ALPH * LX + (1. - ALPH) * NX
192        LET PY = ALPH * LY + (1. - ALPH) * NY
193        LET NPX = NX - PX              LET NPY = NY - PY
194        LET I = RTE.DATA(RT,NM+3*(D.ON.RT-1))
195+       IF FLD.FORM(VEH) NE 0
196+          LET I = FLD.FORM(VEH)
197+       ELSE
198+          IF I EQUALS 0,  LET I = FORMACODE(VEH)
199+          ALWAYS
200+       ALWAYS
201        LET J =     FORMPOS(VEH) * 2
202        LET X.OFF = FORM.OFFSET(I,J-1)
203        LET Y.OFF = FORM.OFFSET(I,J)
204        LET CHG.INT=FOR.CHG.INT(I)
205        LET D.TO.MCP = SQRT.F(NPX*NPX + NPY*NPY)
206        IF D.TO.MCP LESS THAN ZERO.LEVEL
              GO TO MCP.REACHED
363    END
```

4. Routine MOVE.LIMITS

Purpose

Routine MOVE.LIMITS is called from routine MOVE to determine a vehicle's maximum speed, acceleration rate, and deceleration rate. These parameters are a function of the weapon type, the system type, the slope of the terrain and the dismounted tactical situation.

Modifications

The routine now returns speed limits that are also a function of an obstacle encounter speed degradation.

Temporary Attribute Real

FSPEED.FAC

Brief Explanation

Line 18 The effect (degraded speed) of FSPEED.FAC has been introduced here. FSPEED.FAC may have any value between 0 and 1.

CODE

```
 1  ROUTINE MOVE.LIMITS GIVEN VEH,SLOPE YIELDING SPEED,
        ACCEL, DECEL
 2  '' MOVE.LIMITS LOOKS UP LIMITING SPEED AND
        ACCELERATION VALUES BASED ON
 3  ''     SYSTEM TYPE, WEAPON TYPE, AND TERRAIN SLOPE
12     IF SLOPE LE LIM.SPDS(SYS,WPN,2)
13        LET OFFSET = 2     '' DOWN SLOPE
14     ELSE
15        LET OFFSET = 1     '' LEVEL
16     ALWAYS
17  ALWAYS
18+ LET SPEED = LIM.SPDS(SYS,WPN,3+OFFSET)*FSPEED.FAC(VEH)
19  LET ACCEL = LIM.SPDS(SYS,WPN,6+OFFSET)
20  LET DECEL = LIM.SPDS(SYS,WPN,9+OFFSET)
36  END
```

5. Routine RED.CREATE

Purpose

This routine serves the same purpose as does BL.CREATE except it acts for the Red (opposing) force. force.

Modifications

The following four attributes must be initialized:

Temporary Attributes Integer

BLADE.COND
HOLD.FORM
PLOW.COND

Temporary Attribute Real

FSPEED.FAC

CODE

```
 1  ROUTINE RED.CREATE(A,B,MV)
11  CREATE  A  TANK
12  READ NAME(TANK), SYS.TYPE(TANK), WPN.TYPE(TANK),
       SEC(TANK), PLT(TANK),
    CO(TANK),
13  BN(TANK),SQDVEH(TANK),PLTLDR(TANK),COCDR(TANK),
       START.AREA(TANK),
14+ ALIVE.DEAD(TANK),FIR.MODE(TANK),PLOW.COND(TANK),
       BLADE.COND(TANK)
15  CALL FLD.DIST(TANK)
16  LET FLD.INT.DIST(TANK)=RINF.C
17+ LET FSPEED.FAC(TANK)=1.0
18+ LET HOLD.FORM(TANK)=999
19  IF BN(TANK) GT RBNS LET RBNS=BN(TANK) ALWAYS
83  RETURN   END
```

6. Routine RES2

Purpose

Routine RES2 allocates space for n-dimensional arrays. This routine is called from the MAIN routine.

Modifications

The following two arrays have been added to the simulation:

Permanent Attributes Integer

GRUNTLETH (3-D) Integer

This 3-Dimensional array contains the BRL casualty data for anti-personnel mines.

MINLETH (4-D) Integer

This 4-Dimensional array contains the BRL casualty data for anti-tank mines.

CODE

```
 1      ROUTINE RES2
 3+     RESERVE MINLETH AS  5 BY 8 BY 2 BY 4
 4+     RESERVE GRUNTLETH AS  4 BY 2 BY 35
55      RETURN END
```

7. Routine RES5

Purpose

Routine RES5 is called from routine MAIN. This routine reads in data.

Modifications

The following arrays have their data read in.

Permanent Attributes Integer

GRUNTLETH (3-D)
MINLETH (4-D)

CODE

```
 1      ROUTINE RES5
 2      DEFINE I,J,K,L,M AS INTEGER VARIABLES
```

```
  3  USE UNIT 8 FOR INPUT
  4      IF DLA(8) NE 0
  5      FOR I = 1 TO 2 FOR J = 1 TO 2 FOR K =  1 TO 10
  6      READ ADDON(I,J,K)
  7      ALWAYS
  8  FOR I = 1 TO 2 FOR J = 1 TO 4  READ DGNV(I,J)
 9+      FOR I = 1 TO 5 FOR J = 1 TO 8 FOR K = 1 TO 2
             FOR L = 1 TO 4
10+      READ MINLETH(I,J,K,L)
11+      FOR I = 1 TO 4 FOR J = 1 TO 2 FOR K = 1 TO 35
12+      READ GRUNTLETH(I,J,K)
18       RETURN END
```

8. Event TARGET.SELECT

Purpose

Event TARGET.SELECT is called from events DETECT, FIRE, IMPACT, and from itself. This routine selects the highest priority target from an entity's target list for servicing. Selection is based on target range and ammunition availability.

Modifications

This routine has been modified in order to stop a tank from selecting targets and subsequently firing at targets while it is engaged in the following circumstances:

Pushing a dead plow tank through a minefield.

Breaching a gap obstacle with its dozer blade.

Self breaching a gap obstacle.

The following attributes have been added:

Temporary Attribute Integer

BLADE.COND

This unit attribute gives the tank the ability to have a dozer blade characteristic.

0-No blade available.
1-Blade available but not in use.
2-Blade being used or vehicle is self breaching.

PLOW.COND

This unit attribute gives the tank the ability to have a mine plow characteristic.

0-no plow available
1-plow available but not in use
2-plow being used
3-lead vehicle in offset
4-tank that is pushing dead plow tank

TSKED

This attribute serves as a flag to stop further target selects.

Brief Explanation

Lines 20-23 Stops the pushing tank from firing.

Lines 24-27 Stops vehicle that is breaching a gap from firing.

CODE

```
 2  UPON TARGET.SELECT(A)
16  IF FIP(A) = 1
17  RETURN  ELSE
18  IF SCHED(A) = 1
19  RETURN  ELSE
20+ IF PLOW.COND(A) = 4
21+ LET TSKED(A) = 1
22+ SCHEDULE A TARGET.SELECT(A) IN 15 UNITS
23+ RETURN  ELSE
24+ IF BLADE.COND(A) = 2
```

```
25+ LET TSKED(A) = 1
26+ SCHEDULE A TARGET.SELECT(A) IN 15 UNITS
27+ RETURN  ELSE
28  LET FOE(A)=0
29  CALL PURGE.LIST(A)
50  RETURN END
```

9. Routine FLD.ACT

Purpose

Routine FLD.ACT is called from the MOVE routine. This routine sorts out what field actions to perform next. (For a more detailed explanation see ref 3)

Modifications

The following routines are now called from routine FLD.ACT:

GAP.DECISION
GAP.ENTRY
GAP.INTERNAL
GAP.LEAVE
MF.DECISION
MF.ENTRY
MF.EXIT
MF.INTERNAL

These routines are discussed in Appendix C. In order for the appropriate routines to be called the following entity attributes have to be used:

Temporary Attribute Integer

FLD.ART - Code describing the kind of action for pending internal actions.

FLD.NO - Name of the field involved in any pending internal action.

TPNAM.FLD - Field temporary pointer number.

TYP.FLD - Field type code

1 - A mandatory dismount field
2 - A minefield
3 - A minefield decision field
4 - A tank ditch
5 - A road crater
6 - A blown bridge (short-wet gap)
7 - A gap decision field

Brief Explanation

All the additions are self explanatory in the code. They call one of the appropriate routines mentioned above.

Lines 74-78 This change re-initializes attributes that were changed in routine FLD.ACT. It only takes place upon an entities departure from a particular type field (one where its FLD.NO was set to the field pointer number).

```
CODE

 2  ROUTINE FLD.ACT(VEH)  ''SORTS OUT WHICH FIELD
        ACTIONS TO PERFORM
 5  IF FLD.INT.DIST(VEH) LE 0.0
 6    PRINT 1 LINE WITH NAME(VEH),X.CURRENT(VEH),
          Y.CURRENT(VEH),
 7        NAM.FLD(FLD.NO(VEH)),TIME.V AS FOLLOWS
FLD INT ACT VEH=**** LOCN = ****** ****** FIELD = ****
        TIME = ****
 8  ''
 9  '' HERE WE CALL ROUTINES TO PERFORM INTERNAL
        ACTIONS
10  ''
11+   IF FLD.AKT(VEH) = 2
12+     CALL MF.INTERNAL(VEH)
13+   ALWAYS
14+   IF FLD.AKT(VEH) = 4 OR FLD.AKT(VEH) = 5 OR
          FLD.AKT(VEH) = 6
15+     CALL GAP.INTERNAL GIVEN VEH
16+   ALWAYS
17  ALWAYS
```

```
18   IF FLD.BDY.DIST(VEH) LE 0.0
19        LET DX = COS.F(DIR.OF.MVMT(VEH))

39             IF S1 LE 0.0 AND S1 GE -2.0
40               PRINT 1 LINE WITH NAME(VEH),XCUR,YCUR,
                                 NAM.FLD(FIELD),
41                 TIME.V AS FOLLOWS
FIELD ENTRY VEH=**** LOCN = ****** ****** FIELD = ****
          TIME = ****
42   ''
43   '' HERE WE CALL ROUTINES TO PERFORM FIELD ENTRY
                 BOUNDARY ACTIONS
44   ''
45   IF COLOR(VEH)=0 AND WPN.TYPE(VEH)=8 AND
          TYP.FLD(FIELD)=1
46      CALL DISMOUNT(VEH,3)
47   ALWAYS
48+  IF TYP.FLD(FIELD)=2
49+     CALL MF.ENTRY GIVEN VEH, FIELD
50+  ALWAYS
51+  IF TYP.FLD(FIELD) = 3
52+     CALL MF.DECISION(VEH,FIELD)
53+  ALWAYS
54+  IF TYP.FLD(FIELD) = 4 OR TYP.FLD(FIELD) = 5 OR
          TYP.FLD(FIELD) = 6
55+     CALL GAP.ENTRY(VEH,FIELD)
56+  ALWAYS
57+  IF TYP.FLD(FIELD) = 7
58+     CALL GAP.DECISION(VEH,FIELD)
59+  ALWAYS
60             ALWAYS
61             IF S2 LE 0.0 AND S2 GE -2.0
62               PRINT 1 LINE WITH NAME(VEH),XCUR,YCUR,
                        NAM.FLD(FIELD),
63                 TIME.V AS FOLLOWS
FIELD EXIT  VEH = **** LOCN = ****** ****** FIELD = ****
      TIME = ****
64   ''
65   '' HERE WE CALL ROUTINES TO PERFORM FIELD EXIT
             BOUNDARY ACTIONS
66   ''
67+            IF FLD.AKT(VEH) = 2
68+               CALL MF.EXIT GIVEN VEH
69+            ALWAYS
70+            IF FLD.AKT(VEH) = 4 OR FLD.AKT(VEH) = 5
                        OR FLD.AKT(VEH) = 6
71+               CALL GAP.LEAVE GIVEN VEH
72+            ALWAYS
73   ''
74+            IF TPNAM.FLD(FIELD) EQ FLD.NO(VEH)
75+               LET FLD.INT.DIST(VEH) = RINF.C
76+               LET FLD.NO(VEH) = 0
77+               LET FLD.AKT(VEH) = 0
78+            ALWAYS
79          ALWAYS
80        ''ENDIF
81     LOOP
82   ALWAYS
83   CALL FLD.DIST(VEH)
84   RETURN
85   END
```

10. Routine FLD.CREATE

Purpose

Routine FLD.CREATE creates the field entities. This routine is called from the FLD.INIT routine before the start of the battle.

Modifications

Additional field attributes have been added and they needed to be initialized. Several arguments are sent to this routine. Instead of defining them, the attributes that receive their values will be discussed.

Given Arguments (Real)

ANGLE
P5
P6
P7
P8
P9
P10
P11
P12
SAMAJ
SAMIN

Global Variable Integer

FLD.POINTER (1-D)

Contains the pointer values of all the fields.

Temporary Attribute Integer

TPNAM.FLD - Field temporary pointer number

Temporary Attributes Real

ANGLE.FLD - Proper angle in radians from east to major axis.

AREA.FLD - Area of this elliptical field

P5.FLD - Field parameter (Appendix D)

P6.FLD - Field parameter (Appendix D)

P7.FLD - Field parameter (Appendix D)

P8.FLD - Field parameter (Appendix D)

P9.FLD - Field parameter (Appendix D)

P10.FLD - Field parameter (Appendix D)

P11.FLD - Field parameter (Appendix D)

P12.FLD - Field parameter (Appendix D)

SAMAJ.FLD - Semi-major axis length

SAMIN.FLD - Semi-minor axis length

CODE

```
 1    ROUTINE FLD.CREATE
 2  ''CREATE ONE FIELD AND SET ITS ATTRIBUTES
 3+ GIVEN XC,YC,SAMAJ,SAMIN,ANGLE,TYPE,P1,P2,P3,P4,P5,
          P6,P7,P8,P9,P10,
 4+   P11,P12 YIELDING NAME
 5  NORMALLY MODE IS REAL
 6    DEFINE TYPE, NAME, FIELD AS INTEGER VARIABLES
 7    CREATE A FIELD
 8    LET XC.FLD(FIELD) = XC
 9    LET YC.FLD(FIELD) = YC
10    LET TYP.FLD(FIELD) = TYPE
11+   LET ANGLE.FLD(FIELD) = ANGLE
```

```
12+     LET SAMAJ.FLD(FIELD)  = SAMAJ
13+     LET SAMIN.FLD(FIELD)  = SAMIN
14+     LET AREA.FLD(FIELD)  = SAMAJ*SAMIN*PI.C
15+     LET TPNAM.FLD(FIELD)  = FIELD
16      LET P1.FLD(FIELD)  = P1
17      LET P2.FLD(FIELD)  = P2
18      LET P3.FLD(FIELD)  = P3
19      LET P4.FLD(FIELD)  = P4
20+     LET P5.FLD(FIELD)  = P5
21+     LET P6.FLD(FIELD)  = P6
22+     LET P7.FLD(FIELD)  = P7
23+     LET P8.FLD(FIELD)  = P8
24+     LET P9.FLD(FIELD)  = P9
25+     LET P10.FLD(FIELD)  = P10
26+     LET P11.FLD(FIELD)  = P11
27+     LET P12.FLD(FIELD)  = P12
33      ADD 1 TO FLDS.CREATED
34      LET NAM.FLD(FIELD)=FLDS.CREATED
35+     LET FLD.POINTER(NAM.FLD(FIELD)) = FIELD
36      LET NAME = FLDS.CREATED
37      LET SANG = SIN.F(ANGLE)
38      LET CANG = COS.F(ANGLE)
39      LET PXX.FLD(FIELD)  = (CANG/SAMAJ)**2 +
          (SANG/SAMIN)**2
40      LET PYY.FLD(FIELD)  = (SANG/SAMAJ)**2 +
          (CANG/SAMIN)**2
41      LET PXY.FLD(FIELD)  = 2*SANG*CANG*(1/SAMAJ**2 -
          1/SAMIN**2)
42      FILE THIS FIELD IN THE FLD.SET
43      RETURN
44    END
```

11. Routine FLD.INIT

Purpose

Routine FLD.INIT is called once by MAIN before the start of the simulation. It reads the input describing the fields to be created and it calls FLD.CREATE for each such field.

Modifications

The following field attributes have been added to greater enhance the capabilities of an obstacle field. For definitions and input values see Appendix D.

Recursive Variables Real

P5
P6
P7
P8
P9
P10
P11
P12

Global Variables

FLD.POINTER (1-D) INTEGER

This 1-Dimensional array contains the pointer values of all the fields. In this routine the space for array is reserved.

TRAF.CONTROL (3-D) Real (Appendix A)

In this routine the space for this 3-Dimensional array is reserved.

Recursive Variables Integer (Appendix D)

OB.NUM

The number of obstacle fields. It is used to dimension the 3-Dimensional array TRAF.CONTROL.

RT.OB.NUM

The number of routes going through the obstacles. This input variable also dimensions TRAF.CONTROL.

CODE

```
1   ROUTINE FLD.INIT
2  ''CREATE ALL FIELDS THAT ARE IN PLACE AT THE START
        OF THE BATTLE
3    NORMALLY MODE IS REAL
```

```
 4     DEFINE TYPE, NUM, OB.NUM, RT.OB.NUM, I, NAME AS
           INTEGER VARIABLES
 5     USE UNIT 5 FOR INPUT
 6+    READ NUM, OB.NUM, RT.OB.NUM
 7+    RESERVE FLD.POINTER(*) AS NUM
 8+    RESERVE TRAP.CONTROL(*,*,*) AS OB.NUM BY
           RT.OB.NUM BY 5
 9     FOR I = 1 TO NUM DO
10+        READ XC,YC,SAMAJ,SAMIN,ANGLE,TYPE,P1,P2,P3,
               P4,P5,P6,P7,
11+         P8,P9,P10,P11,P12
12         LET ANGLE = ANGLE/RADIAN.C
13+        CALL FLD.CREATE GIVEN XC,YC,SAMAJ,SAMIN,
               ANGLE,TYPE,P1,P2,P3,P4,
14+         P5,P6,P7,P8,P9,P10,P11,P12 YIELDING NAME
15    LOOP
16    RETURN
17  END
```

12. Routine BASIC.LOAD

Purpose

This routine is used to initialize the ammunition basic load levels for all elements.

Modifications

Subscripted labels for the two new vehicle launched bridges had to be added. These additions only serve as exit labels out of this routine.

CODE

```
 1  ROUTINE BASIC.LOAD(A)
 2     DEFINE A,S,W AS INTEGER VARIABLES
 3     LET S = SYS.TYPE(A)   LET W = WPN.TYPE(A)
 4     LET AMMO1(A)  = PJO.CHAR(S,W,1,4)
 5     LET AMMO2(A)  = PJO.CHAR(S,W,2,4)
 6     LET AMMO3(A)  = PJO.CHAR(S,W,3,4)
 7     LET AMMO4(A)  = PJO.CHAR(S,W,4,4)
 8     GO TO SYSTM(S)
19  'SYSTM(1)' GO TO TANKX(WPN.TYPE(A))
20  'TANKX(1)'
21     LET C.1(A) = 16   LET C.2(A) = 6 LET R.CON(A) = 1
       GO OUT
22  'TANKX(2)' LET C.1(A)=CAPDS LET C.2(A)=CHEAT
       LET AP.TOW(A)=CASEAP
23  LET HE.DRAG(A)=CASEHE LET AW1.OR.MSL3(A)=100
24  LET FOM(A)=1 LET VEH.TYPE(A)=1 LET R.CON(A)=1
```

```
25   RETURN
26+  'TANKX(4)'
27+  'TANKX(5)'
28   'TANKX(7)'
29      GO OUT
36   'OUT'
37   'SYSTM(7)'
38      RETURN END
```

APPENDIX C

ROUTINES AND EVENTS FOR THE ENGINEER MODULE IN STAR

This appendix contains the complete documentation for the Engineer Effects Module for the STAR Combat Model.

1. Routine POP.A.MINE

Purpose

Routine POP.A.MINE determines if an entity is within the casualty producing radius of a mine or if the entity has run over a mine. This routine is called from routine MF.INTERNAL.

Given Arguments Integer

TNK- Pointer to the element being moved.

Global Variable Alpha

DAM.ARRAY (1-D)

Contains the possible values for hit.state.

1	NDAM
2	MDAM
3	HDAM
4	MFDM
5	DEAD
6	MISS

Global Variable Integer

DAMAGE.NUM

Indicates the damage status of an entity after having a round impact on or near it.

1	hit but already mf killed
2	mobility damage
3	firepower damage
4	mobility and firepower damage
5	catastrophic kill
6	miss

DEAD.ATKR

Tallies the number of red casualties.

KILLED.MOUNTED

Indicates whether the entity in question was killed while mounted on his vehicle or while dismounted.

ONDISK

Indicates whether or not the user desires the shot list and final attribute list to user specified disk files.

0	no, paper output only
1	yes

YES

Indicates a value of 1. It is used in conditional statements.

Global Variable Real

TARDIM (3-D)

Contains the target dimensions of all entity types in the simulation. This is indexed by SYS.TYPE and WPN.TYPE of the element.

Recursive Variable Integer

DET.LOC

Is the location of the mine detonation in relation to the entity. For anti-personnel frag mines this variable contains the range the entity is from mine detonation. For anti-tank

mines this variable specifies the location of mine detonation on the vehicle.

1 Track hit
2 belly hit

EVEN.BELT

Contains the even mine belt number closest to the actual mine belt to be encountered. This variable is necessary in order to determine how the mines are actually configured on the actual mine belt.

EVEN.N.L.S

Stands for the even number on the left side of the vehicle. This variable helps pin-point the actual mine locations in relation to the entities position on the mine belt.

FLD- Pointer to the field in question.

HALF.BELT

Contains the truncated Integer value when the number of the next encountered mine belt is divided by 2. This is another variable in the sequence used to determine the odd or even numbered representation of belts and mine.

HALF.N.L.S

Contains the truncated integer value that results when the integer number(possible mine location) on the left side of the vehicle is divided by two.

I - Index of a do loop.

J - Subtracts one from the I index.

LEG

Pointer to an element in the platoon of the activating entity (dismounted infantryman).

NUM.LEFT.SIDE

Is the integer location of the left side of the vehicle from the tangent line on the ellipse boundary perpendicular to the semi-major axis.

NUT

Integer mine location under the vehicle/tank.

NXBELT

This local variable receives the value of the entity attribute WHAT.BELT.

WHOCALLED

A flag used by several routines to determine appropriate actions and executions.

Recursive Variable Real

ANGLE

Set to the direction of movement of the entity.

DRAW

Contains the value of a random number from a Uniform(0,1) distribution.

DTOB

Is the distance from the center of a vehicle to the tangent line on the ellipse boundary perpendicular to the semi-major axis.

EFKILL

The probability of at least a firepower kill.

EMKILL

The probability of at least a mobility kill.

EMNFKILL

The union of EMKILL and EFKILL.

KAYKILL

The probability of a catastrophic kill.

SIGMA

Contains the value of the difference between the orientation angle of the field ellipse and the direction of movement of the vehicle.

THETA

Receives the value of the field attribute ANGLE.FLD.

VEH.WIDTH

The width of the vehicle in question.

VW

The vehicle width adjusted for SIGMA that crosses the mine belt.

X.PRIME

The X coordinate location of the entity rotated about orientation angle of the field ellipse.

XMINE

The X coordinate location of the Anti-personnel frag mine.

YMINE

The Y coordinate location of the Anti-personnel frag mine.

Routines called

ATRIT

A routine which assesses the attrition against a Target which has been hit. It checks for a catastrophic kill, increments mobility and firepower damage, and checks the FKILL, MKILL, and MFKILL levels.

FIND.A.MINE (this appendix)

PROB.MINE (this appendix)

Sets Used

PLT.UNIT (1-D)

This set is owned by the permanent entity PLATOON LEADER. It contains the list of temporary entities owned by this platoon.

Temporary Attribute Alpna

HIT.STATE

An alpha variable indicating whether or not an element is alive or dead.

Temporary Attribute Integer

ALIVE.DEAD

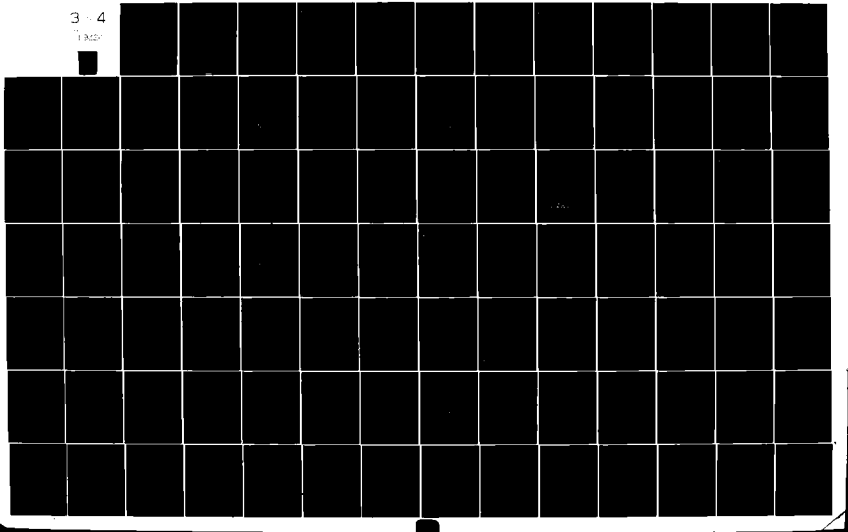
Indicates whether the entity is alive or dead.

0 alive
1 dead
2 alive mounted in carrier.

AD-A119 326 NAVAL POSTGRADUATE SCHOOL MONTEREY CA F/G 15/7
THE ENGINEER EFFECTS MODULE FOR THE STAR COMBAT MODEL.(U)
MAR 82 S C MAIN, J V MUDD

UNCLASSIFIED NL

3 - 4



PLOW.COND

This unit attribute gives the tank the ability to have a mine plow characteristic.

COLOR

Indicates the color of the element.

0	red (attacker)
1	blue (defender)

DEFNUM

The current position or activity of an element.

1	full defilade
2	turret defilade
3	firing defilade
4	half vehicle defilade
5	moving (defilade determined by the Terrain model)
6	reached final area in movement

FIRED.AT

Indicates the total number of rounds fired at a entity.

FKILL

Indicates whether an entity has sustained a firepower kill.

0	no
1	yes

FLD.NO

The vehicle attribute that Contains the name of the field involved in any pending internal action.

K.HIT

Indicates the number of hits sustained by an entity which were sufficient to cause a catastrophic kill.

KKILL

Indicates whether an entity has sustained a catastrophic kill.

0	no
1	yes

MFKILL

Indicates whether an entity has sustained a simultaneous mobility and firepower kill.

0	no
1	yes

MKILL

Indicates whether an entity has sustained a mobility kill.

0	no
1	yes

NAME

The element number of the entity.

NUM.HIT

Indicates the total number of hits sustained by an entity. This includes no damage hits.

PLT

The number of the platoon to which the entity belongs.

SYS.TYPE

This represents the general class of the system of the entity.

1	Tanks
2	mounted infantry
3	dismounted infantry
4	artillery
5	air
6	air defense
7	bunker

8 comm/ew/acq/intel
9 other

WHAT.BELT

This entity attribute keeps the updated belt number that is to be encountered next.

WPN.TYPE

Describes the specific system within the system code. for example, system type 1 is a tank and weapon type four for this system indicates an AVLB.

Temporary Attribute Real

ANGLE.FLD

Orientation angle in radians measured counterclockwise from east to the major axis of the elliptical field.

DIR.OF.MVMT

Indicates the entity's direction of movement measured in radians from east.

F.D

Indicates the accumulated percentage of firepower damage sustained by the entity.

M.D

Indicates the accumulated percentage of mobility damage sustained by the entity.

P3.FLD

Mine Type (1-11)

1 M70 Scatterable Anti-tank mine
2 M56 Scatterable Anti-tank mine
3 M21 Hand-emplaced Anti-tank mine

4	M15 Hand-emplaced Anti-tank mine
5	M19 Hand-emplaced Anti-tank mine
6	M14 Anti-personnel blast mine
7	M25 Anti-personnel blast mine
8	M16 Anti-personnel frag mine
9	ADAM Anti-personnel frag mine
10	M74 Anti-personnel frag mine
11	Claymore Anti-personnel frag mine

P4.FLD (Appendix D)

SAMAJ.FLD

The semi-major axis length of the elliptical field.

SPD

The entity's speed at the end of the most recent movement update.

X.CURRENT

The X-coordinate for the entity as of the last movement update.

XC.FLD

The X-coordinate of the center of an elliptical field.

Y.CURRENT

The Y-coordinate for the entity as of the last movement update.

Z.CURRENT

The elevation for the entity as of the last movement update.

Brief Description

Lines 8-11	Determines if the mine or mine belt was detected and avoided.
Lines 12-42	Subjects dismounted infantry to the effects of Anti-personnel mines if applicable.
Lines 14-17	Subjects dismounted infantry to the effects of Anti-personnel blast mines.
Lines 18-41	Subjects dismounted infantry to the effects of Anti-personnel frag mines.
Lines 43-103	Subjects vehicles to the effects of scatterable and belted/patterned Anti-tank mines.
Lines 43-54	Subjects vehicles to the effects of scatterable Anti-tank mines. The detonation location under the vehicle is also determined.
Lines 55-102	Subjects the vehicle to the effects of a belt of mines. A vehicle may receive more than one hit at a time. The exact detonation location under a

vehicle is determined because the mines are explicitly played.

Lines 105-128 Updates the killer-victim score board and it prints out a shot/mine record.

CODE

```
1     ROUTINE POP.A.MINE GIVEN TNK
2     DEFINE TNK,DET.LOC,WHOCALLED,FLD,NUM.LEFT.SIDE,
             HALF.BELT,NXBELT,EVEN.BELT,HALF.N.L.S,
3            EVEN.N.L.S,I,J,HOT,LEG AS INTEGER
             VARIABLES
4     DEFINE EMKILL, EPKILL, KAYKILL, DRAW,THETA,
             X.PRIME,DTOB,ANGLE,SIGMA,EMNPKILL,
5            VEH.WIDTH,VW,XMINE,YMINE AS REAL
             VARIABLES
6     LET WHOCALLED = 1
7     LET FLD = FLD.NO(TNK)
8     LET DRAW=UNIFORM.F(0.,1.,7)
9     IF P4.FLD(FLD) GT DRAW  ''DETECT AND AVOID
10      RETURN
11    OTHERWISE
12 IF SYS.TYPE(TNK) = 3 ''DISMOUNTED INFANTRY
13    IF P3.FLD(FLD) GE 6.0
14       IF P3.FLD(FLD) LE 7.0  ''AP-BLAST MINES
15          CALL PROB.MINE GIVEN TNK, DET.LOC
16                YIELDING EMKILL,EPKILL,EMNPKILL,
                                              KAYKILL
17          CALL ATRIT(TNK,TNK,EMKILL,EPKILL,EMNPKILL,
                                   KAYKILL,WHOCALLED)
18       ELSE''AP-FRAG MINES
19          CALL FIND.A.MINE GIVEN TNK YIELDING XMINE,
                                              YMINE
20          FOR EVERY LEG IN PLT.UNIT(PLT(TNK)) WITH
                          ALIVE.DEAD(LEG) NE 2 AND
21                           SYS.TYPE(LEG) EQ 3, DO
22             LET DET.LOC =
                INT.F(SQRT.F((X.CURRENT(LEG)-XMINE)**2+
23                           (Y.CURRENT(LEG)-YMINE)**2))
24             IF DET.LOC = 0
25               ADD 1 TO DET.LOC
26             ALWAYS
27             IF P3.FLD(FLD) = 11.0  ''CLAYMORE MINE
28               IF DET.LOC GT 116
29                 CYCLE
30               OTHERWISE
31               ADD 24 TO DET.LOC ''TO MAKE UP FOR
                                          SAHAJ AXIS
32             ELSE
33               IF DET.LOC GT 35
34                 CYCLE
35               OTHERWISE
36             ALWAYS
37             CALL PROB.MINE GIVEN TNK, DET.LOC
38                   YIELDING EMKILL,EPKILL,EMNPKILL,
                                              KAYKILL
39             CALL ATRIT(TNK,TNK,EMKILL,EPKILL,
                          EMNPKILL,KAYKILL,WHOCALLED)
```

```
40              LOOP
41           ALWAYS
42        ALWAYS
43     ELSE  ''MOTORIZED VEHICLE
44        IF P3.FLD(FLD) LT 6
45           IF P3.FLD(FLD) LT 3   '' SCATTERABLE MINE FIELD
46              LET DRAW=UNIFORM.F(0.,1.,7)
47              IF DRAW LE (.3333)
48                 LET DET.LOC = 1
49              ELSE
50                 LET DET.LOC = 2
51              ALWAYS
52              CALL PROB.MINE GIVEN TNK, DET.LOC
53                        YIELDING EMKILL,EFKILL,EMNPKILL,
                                                  KAYKILL
54              CALL ATRIT(TNK,TNK,EMKILL,EFKILL,EMNPKILL,
                                          KAYKILL,WHOCALLED)
55           ELSE
56              LET NXBELT=WHAT.BELT(TNK)
57              LET THETA = ANGLE.FLD(FLD)
58              LET X.PRIME = X.CURRENT(TNK)*COS.F(THETA)+
                              Y.CURRENT(TNK)*SIN.F(THETA)
59              ''DTOB =DISTANCE TO THE OUTER BOUNDRY
60              LET DTOB = SAMAJ.FLD(FLD)
                              -ABS.F(XC.FLD(FLD)-X.PRIME)
61              IF DIR.OF.MVMT(TNK) LT 0.0
62                 LET ANGLE = 360/RADIAN.C +
                                          DIR.OF.MVMT(TNK)
63              ELSE
64                 LET ANGLE = DIR.OF.MVMT(TNK)
65              ALWAYS
66                 LET SIGMA = ANGLE - THETA
67                 LET VEH.WIDTH =
                      TARDIM(SYS.TYPE(TNK),WPN.TYPE(TNK),6)
68                 LET VW = VEH.WIDTH/
                         COS.F(ABS.F((90/RADIAN.C)-SIGMA))
69                 LET NUM.LEFT.SIDE = TRUNC.F(DTOB-VW/2)-1
70                 LET HALF.BELT = TRUNC.F(NXBELT/2)
71                 LET EVEN.BELT = HALF.BELT*2
72                 LET HALF.N.L.S =
                                  TRUNC.F(NUM.LEFT.SIDE/2)
73                 LET EVEN.N.L.S =HALF.N.L.S*2
74                 IF NXBELT EQ EVEN.BELT
75                    IF EVEN.N.L.S EQ NUM.LEFT.SIDE
76                       SUBTRACT 1 FROM NUM.LEFT.SIDE
77                    ALWAYS
78                 ELSE
79                    IF EVEN.N.L.S NE NUM.LEFT.SIDE
80                       SUBTRACT 1 FROM NUM.LEFT.SIDE
81                    ALWAYS
82                 ALWAYS
83                    FOR I = 1 TO 10, DO
84                    LET J = I-1
85                    LET NUT = J*2.0+NUM.LEFT.SIDE
                            ''NUT = NUMBER.UNDER.TANK
86                    IF NUT GT (DTOB+VW/2)
87                       GO TO 'PRINT'
88                    OTHERWISE
89                    IF NUT LT (DTOB-VW/2)
90                       CYCLE
91                    OTHERWISE
92                    IF NUT LT (DTOB+VW/3) AND
                                          NUT GT (DTOB-VW/3)
93                       LET DET.LOC = 2
94                    ELSE
95                       LET DET.LOC = 1
96                    ALWAYS
97                       CALL PROB.MINE GIVEN TNK, DET.LOC
```

```
98                                  YIELDING EMKILL,EFKILL,
                                             EMNFKILL,KAYKILL
99                  CALL ATRIT(TNK,TNK,EMKILL,EFKILL,
100                             EMNFKILL,KAYKILL,WHOCALLED)
101                   LOOP
102       ALWAYS
103     ALWAYS
104   ALWAYS
105   'PRINT'  LET HIT.STATE(TNK) = DAM.ARRAY(DAMAGE.NUM)
106   USE UNIT 6 FOR OUTPUT
107      START NEW LINE
108   'UP'
109   WRITE NAME(TNK), WPN.TYPE(TNK), TIME.V,
      X.CURRENT(TNK), Y.CURRENT(TNK),
110   Z.CURRENT(TNK), SPD(TNK), DEFNUM(TNK),
      HIT.STATE(TNK), M.D(TNK),
111   F.D(TNK), MKILL(TNK), FKILL(TNK), MFKILL(TNK),
112   KKILL(TNK), K.HIT(TNK), FIRED.AT(TNK), NUM.HIT(TNK)
113   AS S 6, "MINE", S 12, 2 I 4, I 5, S 5, S 22, 3 I 7,
         D(4,1), I 3,
         S 5, S 6, S 1, A 4, 2 D(5,2), 2 I 2, 5 I 3
114   IF KILLED.MOUNTED NE 0
115   ALWAYS
116   LET KILLED.MOUNTED=0
117   IF ONDISK=YES AND WRITE.V=6 USE UNIT 10 FOR OUTPUT
                                 START NEW LINE GO TO UP ELSE
118     IF COLOR(TNK) EQ 0  AND ( DAMAGE.NUM EQ 5 OR
                                    ((DAMAGE.NUM EQ 4 OR
119       DAMAGE.NUM EQ 3 OR DAMAGE.NUM EQ 2 ) AND
120    (    MFKILL(TNK) EQ 1)))
121    AND ( DAMAGE.NUM NE 5 OR MFKILL(TNK) NE 1 )
122      AND HIT.STATE (TNK) NE "DEAD" AND
                                 HIT.STATE(TNK) NE "ADED"
123   ADD 1 TO DEAD.ATKR ALWAYS
124   IF HIT.STATE(TNK) NE "DEAD" AND
                                 HIT.STATE(TNK) NE "ADED"
125   LET HIT.STATE(TNK)="    "   ALWAYS
126   START NEW LINE
127      RETURN END
```

2. Routine FIND.A.MINE

Purpose

Routine FIND.A.MINE finds the X,Y coordinate locations of the Anti-personnel frag mines in question. This routine is called from Routine POP.A.MINE.

Given Arguments Integer

B - Pointer to the element being moved.

Yielding Arguments Real

XMINE

The X coordinate location of the Anti-personnel frag mine.

YMINE

The Y coordinate location of the Anti-personnel frag mine.

Recursive Variable Integer

FLD

Name of the field involved in this pending action. It serves as a pointer to access other field attributes.

Recursive Variable Real

ANGLE

Set to the direction of movement of the entity.

DRAW

Contains the value of a random number from a Uniform(0,1) distribution.

MYN.DIST

The distance in meters that the activating entity is from the Anti-personnel frag mine.

PSI

The direction of movement of the entity minus 90 degrees.

XDELTA

The distance in the X direction from the activating entity's location to the frag mine. The user may think of the MYN.DIST as a vector perpendicular to the direction of movement.

YDELTA

The distance in the Y direction from the activating entity's location to the frag mine. The user may think of the MYN.DIST as a vector perpendicular to the direction of movement.

Temporary Attribute Integer

FLD.NO

The vehicle attribute that Contains the name of the field involved in any pending internal action.

Temporary Attribute Real

DIR.OF.MVMT

Indicates the entity's direction of movement measured in radians from the east.

P2.FLD

This field attribute contains the length of the frag mine trip wire in meters.

P3.FLD

Mine Type (1-11)

1	M70 Scatterable Anti-tank mine
2	M56 Scatterable Anti-tank mine
3	M21 Hand-emplaced Anti-tank mine
4	M15 Hand-emplaced Anti-tank mine
5	M19 Hand-emplaced Anti-tank mine
6	M14 Anti-personnel blast mine
7	M25 Anti-personnel blast mine
8	M16 Anti-personnel frag mine
9	ADAM Anti-personnel frag mine
10	M74 Anti-personnel frag mine
11	Claymore Anti-personnel frag mine

X.CURRENT

The X-coordinate for the entity as of the last movement update.

XC.FLD

The X-coordinate of the center of an elliptical field.

Y.CURRENT

The Y-coordinate for the entity as of the last movement update.

YC.FLD

The Y-coordinate of the center of an elliptical field.

Brief Explanation

Lines 6-8 Sets the X,Y coordinate location of a Claymore mine to that of the center of the elliptical field.

Lines 10-39 Sets the X,Y coordinate location of the fragmentation mine on a Line that is perpendicular to the entity's direction of movement. The final disposition of the mine in relation to the activating entity is found through a Monte Carlo process.

CODE

```
 1  ROUTINE FIND.A.MINE GIVEN B YIELDING XMINE,YMINE
 2  DEFINE B,FLD AS INTEGER VARIABLES
 3  DEFINE XMINE,YMINE,DRAW,MYN.DIST,ANGLE,PSI,XDELTA,
 4         YDELTA AS REAL VARIABLES
 5  LET FLD = FLD.NO(B)
 6  IF P3.FLD(FLD) = 11.0  ''CLAYMORE MINE
 7     LET XMINE = XC.FLD(FLD)
 8     LET YMINE = YC.FLD(FLD)
 9  ELSE  ''OTHER TYPE OF AP.FRAG MINE
10     LET DRAW = UNIFORM.F(0.,1.,7)
11     LET MYN.DIST = .707*P2.FLD(FLD)*DRAW
12     LET ANGLE = ABS.F(DIR.OF.MVMT(B))
13     LET PSI = ANGLE-(90.0/RADIAN.C)
14     IF DIR.OF.MVMT(B) LT 0.0
15        IF PSI LT 0.0
16           LET PSI = ABS.F(PSI)
17        ELSE
18           LET PSI = (-1.0)*PSI
19        ALWAYS
20     ALWAYS
21     LET XDELTA = MYN.DIST*COS.F(ABS.F(PSI))
22     LET YDELTA = MYN.DIST*SIN.F(ABS.F(PSI))
23     IF DRAW LE .5  ''MINE ON RIGHT OF VEHICLE
24        IF PSI LT 0.0
25           LET XMINE = X.CURRENT(B)+XDELTA
26           LET YMINE = Y.CURRENT(B)-YDELTA
27        ELSE
28           LET XMINE = X.CURRENT(B)+XDELTA
29           LET YMINE = Y.CURRENT(B)+YDELTA
30        ALWAYS
31     ELSE  '' MINE ON LEFT OF VEHICLE
32        IF PSI LT 0.0
33           LET XMINE = X.CURRENT(B)-XDELTA
34           LET YMINE = Y.CURRENT(B)+YDELTA
35        ELSE
```

```
36              LET XMINE = X.CURRENT(B)-XDELTA
37              LET YMINE = Y.CURRENT(B)-YDELTA
38          ALWAYS
39      ALWAYS
40  ALWAYS
41  RETURN
42  END
```

3. Routine PROB.MINE

Purpose

Routine PROB.MINE accesses the appropriate casualty data from the arrays MINLETH and GRUNTLETH. This routine is called from routine POP.A.MINE.

Given Arguments Integer

DET.LOC

Is the location of the mine detonation in relation To the entity. For Anti-personnel frag mines this variable contains the range the entity is from mine detonation. For Anti-tank mines this variable specifies the location of mine detonation on the vehicle.

1 Track hit
2 belly hit

TNK- Pointer to the element being moved.

Yielding Arguments Real

EFKILL

The probability of at least a firepower kill.

EMKILL

The probability of at least a mobility kill.

EMNFKILL

The union of EMKILL and EFKILL.

KAYKILL

The probability of a catastrophic kill.

Permanent Attribute Integer

GRUNTLETH (3-D)

This 3-dimensional array Contains the BRL casualty data for Anti-personnel mines.

MINLETH (4-D)

This 4-dimensional array Contains the BRL casualty data for Anti-tank mines.

Recursive Variable Integer

JO

The pointer used to access the appropriate mine casualty data based on vehicle type for Anti-tank mines and firing position for Anti-personnel mines.

MYN.TYPE

This is the pointer used to access Anti-personnel mine casualty data based on appropriate mine type.

TYPE

This variable contains the value of P3.FLD, the mine type in question.

Recursive Variable Real

DRAW

Contains the value of a random number from a Uniform(0,1) distribution.

MAYKILL

Contains the appropriate probability of kill accessed from GRUNTLETH.

Temporary Attribute Integer

DEFNUM

The current position or activity of an element.

1	full defilade
2	turret defilade
3	firing defilade
4	half vehicle defilade
5	moving (defilade determined by the terrain model)
6	reached final area in movement

FLD.NO

Name of the field involved in any pending internal action.

SYS.TYPE

This represents the general class of the system of the entity.

1	tanks
2	mounted infantry
3	dismounted infantry
4	artillery
5	air
6	air defense
7	bunker
8	comm/ew/acq/intel
9	other

Temporary Attribute Real

P3.FLD

Mine Type (1-11)

1	M70 Scatterable Anti-tank mine
2	M56 Scatterable Anti-tank mine
3	M21 Hand-emplaced Anti-tank mine
4	M15 Hand-emplaced Anti-tank mine
5	M19 Hand-emplaced Anti-tank mine
6	M14 Anti-personnel blast mine
7	M25 Anti-personnel blast mine
8	M16 Anti-personnel frag mine
9	ADAM Anti-personnel frag mine
10	M74 Anti-personnel frag mine
11	Claymore Anti-personnel frag mine

Brief Explanation

Lines 6-18 Access the Anti-tank mine BRL data from MINLETH.

Lines 19-51 Access the Anti-personnel mine data from GRUNTLETH. This data is in the form of kill probabilities and must be Monte Carlo'ed against to determine casualties.

CODE

```
 1  ROUTINE PROB.MINE GIVEN TNK, DET.LOC
 2          YIELDING EMKILL,EFKILL,EMNFKILL,KAYKILL
 3  DEFINE TNK, TYPE,DET.LOC,JO,MYN.TYPE
          AS INTEGER VARIABLES
 4  DEFINE EMKILL,EFKILL,EMNFKILL,KAYKILL,MAYKILL,
          DRAW AS REAL VARIABLES
 5  LET TYPE= P3.FLD(FLD.NO(TNK))
 6  IF TYPE LT 6.0   ''ANTI-TANK MINES
 7  ''WE NEED SOME WAY TO DETERMINE JO-VEHICLE TYPES
 8     IF SYS.TYPE(TNK) NE 3
 9        IF SYS.TYPE(TNK) = 1
10           LET JO = 1
11        ELSE
12           LET JO = 2
13        ALWAYS
14     ALWAYS
15     LET EMKILL=MINLETH (TYPE,JO,DET.LOC,1) /100.
16     LET EFKILL=MINLETH (TYPE,JO,DET.LOC,2) /100.
17     LET EMNFKILL=MINLETH (TYPE,JO,DET.LOC,3) /100.
18     LET KAYKILL=MINLETH (TYPE,JO,DET.LOC,4) /100.
19  ELSE   ''ANTI-PERSONNEL MINES
20     IF TYPE LT 8.0  ''AP-BLAST
21        LET EMKILL = 1.0
22        LET EFKILL = 1.0
23        LET EMNFKILL = 1.0
24        LET KAYKILL = 1.0
25     ELSE  ''AP-FRAG
26        IF DEFNUM(TNK) = 5''THE SOLDIER IS STANDING
27           LET JO = 1  ''STANDING
28        ELSE  ''THE SOLDIER IS IN THE PRONE POSTION
29           LET JO = 2  ''PRONE
30        ALWAYS
31        IF TYPE EQ 11 ''CLAYMORE MINE
32          LET DET.LOC = INT.F(DET.LOC/10.0)
33           IF DET.LOC = 0
34              ADD 1 TO DET.LOC
35           ALWAYS
36        ALWAYS
37        LET MYN.TYPE = TYPE - 7
```

```
38          LET MAYKILL =
                        GRUNTLETH(MYN.TYPE,JO,DET.LOC)/100.0
39          LET DRAW = UNIFORM.F(0.,1.,7)
40          IF MAYKILL GT DRAW   ''THE SOLDIER DIES
41             LET EMKILL = 1.0
42             LET EFKILL = 1.0
43             LET EMNFKILL = 1.0
44             LET KAYKILL = 1.0
45          ELSE   ''THE SOLDIER LIVES TO FIGHT ANOTHER DAY
46             LET EMKILL = 0.0
47             LET EFKILL = 0.0
48             LET EMNFKILL = 0.0
49             LET KAYKILL = 0.0
50          ALWAYS
51       ALWAYS
52    ALWAYS
53    RETURN
54    END
```

4. Routine MF.DECISION

Purpose

Routine MF.DECISION allows the entity and his platoon the options of going to the minefield to attempt a "bull through" or breach, and the option to bypass or lane select, whichever is applicable. This routine is called from routine FLD.ACT.

Argument Integer

A - Pointer to the element being moved.

FLD- Pointer to the field in question.

Events Scheduled

DIVERT (this appendix)

QUIK.MOVE (this appendix)

TURN.AROUND (this appendix)

Global Variable

FLD.POINTER (1-D) INTEGER

This 1-dimensional array contains the temporary field pointers. This enables the user to access the obstacles by using the sequential order numbers given the fields on input.

TRAP.CONTROL (3-D) REAL (Appendix A)

Recursive Variable Integer

COUNTER

Contains the number of five second move increments To get from the decision ellipse through the obstacle under the worst of conditions.

I - Index for a do loop.

KNOW.LEVEL

Contains the value of the field attribute P6.FLD. This is the state of intelligence known about the obstacle.

0	No knowledge exists
2	knowledge exists
3	knowledge exists and a lane exists

OBS.FLD

The temporary field pointer of the obstacle.

OBS.NUM

The number of the obstacle to our front. This Number is gotten from the field's sequence on input.

STAT.PLOW

This variable gives the platoons status of mine plows.

0-no plow available
1-plow available but not in use
2-plow being used
3-platoon is offsetting
4-platoon is pushing a dead plow tank

Recursive Variable Real

DEL.TIME

Receives the plow activation delay time from the minefield attribute P9.FLD.

DIST.AWAY

This is the distance to the obstacle from this decision ellipse.

INTERV

A constant term that is set so that element locations are updated every five seconds during a small portion of the simulation.

PUSH.FACTOR

Receives the value of the minefield attribute P11.FLD, the push dead plow speed degradation factor.

SMAJ.OBS

Receives the value in meters of the semi-major axis of the minefield ellipse to the front.

Routines Called

PLW.ALIVE (this appendix)

Temporary Attribute Integer

BYPASS

This variable is an argument for Event DIVERT. It carries the route number that will cause lateral movement.

DIRC.ON.RT

Indicates whether or not a vehicle is moving forward or backward on his route.

0	Vehicle is moving in order of increasing MCP numbers along the route. (forward)
1	Vehicle is moving in order of decreasing MCP numbers along the route. (backward)

END.AREA

The number which identifies the ending movement area for this entity.

FLD.AKT

Code describing the kind of action for pending internal actions.

FLD.NO

Name of the field involved in any pending internal action.

FORMACODE

The formation number of the formation to be used by the platoon.

0 Not in formation, vehicle moves along route without offset.

HOLD.FORM

This unit attribute is a storage place, where the units formation number for the platoon (FORMACODE) can be placed when lateral route movement is appropriate.

HOME

This variable is an argument for event DIVERT. It carries the MCP number that will be cause lateral movement.

NEXT.MCP

Movement control point number (on the designated route) toward which the element is now moving.

0 end of route has been reached

NUM.POINTER

This variable is an argument for event QUIK.MOVE. It carries the number of the entity to be moved.

REP.NUMBER

This variable is an argument for event QUIK.MOVE. It carries the number of iterations of scheduling left for this event to perform.

ROUTE

Indicates the number of the route along which the element is travelling.

0 Not using a route

SIDE.STEPPER

This variable is an argument for event DIVERT. It carries the number of the entity to be moved laterally.

START.AREA

The number which identifies the starting movement area for this entity.

TWIST

This variable is an argument for event TURN.AROUND. It carries the number of the entity to be turned in the opposite direction.

TYP.FLD
Field Type Code

1 - a mandatory dismount field
2 - a minefield
3 - a minefield decision field
4 - a tank ditch
5 - a road crater
6 - a blown bridge (short-wet gap)
7 - a gap decision field

Temporary Attribute Real

FLD.INT.DIST

This attribute of an entity contains the distance to a pending field internal action.

INT.TIME

This variable is an argument for event QUIK.MOVE. It carries the time interval number when the next event is to be performed.

P1.FLD (Appendix D)

P2.FLD (Appendix D)

P3.FLD (Appendix D)

P4.FLD (Appendix D)

P5.FLD (Appendix D)

P6.FLD (Appendix D)

P9.FLD (Appendix D)

P11.FLD (Appendix D)

SAMAJ.FLD

The semi-major axis length of the elliptical field.

SPD

The entity's speed at the end of the most recent movement update.

XC.FLD

The X-coordinate of the center of an elliptical field.

YC.FLD

The Y-coordinate of the center of an elliptical field.

Brief Explanation

Lines 11-14 Switches the values of HOLD.FORM and FORMACODE. This allows the entity to pick up the route formation if applicable.

Lines 17-20 Checks if this decision ellipse is on a bypass around the minefield and takes appropriate actions if it is.

Lines 21-22 Links to the knowledge attribute, *P6.FLD*, of the minefield to the front.

Lines 23-24 Turns the entity around if he is coming back from the minefield.

Lines 26-28 Allows the user to stipulate if a bypass or a "bull through" tactic is going to be used.

Lines 31-40 Sets the counter variable to the appropriate number of QUIK.MOVE

iterations needed to get through the minefield.

Line 41 Checks on the status of the plows in the platoon.

Lines 42-46 Determines if the platoon has no knowledge of the minefield or if they have an alive plow, and allows them to proceed toward the minefield.

Lines 47-60 Determines if this minefield has a lane in it. If it does, it allows the entity to move laterally to get to it.

Lines 61-63 Allows the platoon that knows the minefield exists to bypass the minefield.

CODE

```
 1  ROUTINE MF.DECISION GIVEN A, FLD
 2  DEFINE A,FLD,OBS.NUM,OBS.FLD,KNOW.LEVEL,COUNTER,
 3         STAT.PLOW,I AS INTEGER VARIABLES
 4  DEFINE DEL.TIME,PUSH.FACTOR,SMAJ.OBS,INTERV,
 5         DIST.AWAY AS REAL VARIABLES
 6  LET FLD.NO(A)=FLD
 7  LET FLD.AKT(A)=TYP.FLD(FLD)
 8  LET P4.FLD(FLD)= ROUTE(A)
 9  LET P5.FLD(FLD)= NEXT.NCP(A)
10  LET INTERV = 5
11  IF HOLD.FORM(A) NE 999
12     LET FORMACODE(A) = HOLD.FORM(A)
13     LET HOLD.FORM(A) = 999
14  ALWAYS
15  ''*******LINK TO OBSTACLE KNOWLEDGE ATTRIBUTE*******
16  LET OBS.NUM = INT.F(P3.FLD(FLD))
17  IF OBS.NUM = 0
''THIS ELLIPSE IS ON A BYPASS ROUTE-NO OBSTACLE TO FRONT
```

```
18      LET FLD.INT.DIST(A) = RINF.C
19      RETURN
20    OTHERWISE
21    LET OBS.FLD = FLD.POINTER(OBS.NUM)
22    LET KNOW.LEVEL = INT.F(P6.FLD(OBS.FLD))
23    IF DIRC.ON.RT(A) = 1
24      SCHEDULE A TURN.AROUND GIVEN A NOW
25    ELSE
26      IF P1.FLD(FLD) NE 0.0 '' BYPASS EXISTS
27        LET TRAF.CONTROL(OBS.NUM,ROUTE(A),2) =
                                                NEXT.MCP(A)
28      ALWAYS
29    ALWAYS
30    LET FLD.INT.DIST(A) = RINF.C
31    IF P6.FLD(FLD) LE 0
32       LET DEL.TIME = P9.FLD(OBS.FLD)
33       LET PUSH.FACTOR = P11.FLD(OBS.FLD)
34       LET SMAJ.OBS = SAMAJ.FLD(OBS.FLD)
35       LET DIST.AWAY =
                  SQRT.F((XC.FLD(FLD)-XC.FLD(OBS.FLD))**2+
36                       (YC.FLD(FLD)-YC.FLD(OBS.FLD))**2)
37       LET COUNTER =
              INT.F((DEL.TIME+INTERV+(DIST.AWAY/SPD(A))+
38            ((2*SMAJ.OBS)/(PUSH.FACTOR*SPD(A))))/INTERV)
39       LET P6.FLD(FLD) = COUNTER
40    ALWAYS
41    CALL PLW.ALIVE GIVEN A YIELDING STAT.PLOW
42    IF KNOW.LEVEL LE 1 OR STAT.PLOW EQ 1
43      LET COUNTER = INT.F(P6.FLD(FLD))
44      SCHEDULE A QUIK.MOVE GIVEN A,COUNTER,INTERV
                                          IN INTERV UNITS
45      RETURN
46    OTHERWISE
47    IF KNOW.LEVEL = 3
48      IF TRAF.CONTROL(OBS.NUM,ROUTE(A),3) NE RINF.C
49        FOR I = 1 TO DIM.F(TRAF.CONTROL(OBS.NUM,*,*)),
                                                    DO
50          IF TRAF.CONTROL(OBS.NUM,I,3) EQ RINF.C AND
51                (TRAF.CONTROL(OBS.NUM,I,2) NE 0.0 AND
52                 TRAF.CONTROL(OBS.NUM,I,2) LT 1100.0)
53              SCHEDULE A DIVERT(A,
                      INT.F(TRAF.CONTROL(OBS.NUM,I,1)),
54                  INT.F(TRAF.CONTROL(OBS.NUM,I,2))) NOW
55              RETURN
56          OTHERWISE
57        LOOP
          ''FUZZY KNOWLEDGE-LANE EXISTS-NOT PRIVY TO IT
58        LET KNOW.LEVEL = 2
59      ALWAYS
60    ALWAYS
61    IF KNOW.LEVEL = 2 AND P1.FLD(FLD) NE 0.0
62      SCHEDULE A DIVERT(A,INT.F(P1.FLD(FLD)),
                                    INT.F(P2.FLD(FLD))) NOW
63    ALWAYS
64    RETURN
65    END
```

5. Routine MF.ENTRY

Purpose

Routine MF.ENTRY allows an entity to take the appropriate actions when encountering a minefield. This routine is called from routine FLD.ACT.

Given Arguments Integer

A - Pointer to the entity that has entered the minefield.

FLD- Pointer to the field in question.

Events Scheduled

QUIK.MOVE (this appendix)

Global Variable Real

TRAF.CONTROL (3-D) (Appendix A)

Permanent Attribute Integer

PLT.COND (1-D)

This attibute of the PLATOON LEADER keeps track of the present action status of the platoon in a minefield. (i.e. plowing, offsetting, pushing, etc.)

Recursive Variable Integer

BUMP.NUM

Receives the number of move increments that have been made clearing the appropriate route in question. Its value is stored in the TRAF.CONTROL array.

COUNTER

Contains the number of five second move increments that will take place for this entity. This movement update is accomplished by scheduling event QUIK.MOVE.

INCREMENT

Contains the number of move increments that this entity has to make to keep up with the lead element of his platoon. This number serves as a flag to determine if there is remaining cleared movement to our front or if Routine MINE.SCHED has to be called.

KNOW.LEVEL

Contains the value of the field attribute P6.FLD. This is the state of intelligence known about the obstacle.

0	No knowledge exists
2	knowledge exists
3	knowledge exists and a lane exists

LEG

Pointer to an element in the platoon of the activating entity.(dismounted infantryman)

OBS.NUM

Receives the value of the field attribute NAM.FLD. This is the sequential id number in order of field creation.

STAT.PLOW

This variable gives the platoons status of mine plows.

0-no plow available
1-plow available but not in use
2-plow being used
3-platoon is offsetting
4-platoon is pushing a dead plow tank

Recursive Variable Real

CUM.DIST.CLEAR

This is the cumulative distance clear on a route through an obstacle. It receives its value from the TRAF.CONTROL array.

INTERV

A constant term that is set so that element locations are updated every five seconds during a small portion of the simulation.

MF.DET.DIST

This is the distance that a entity will travel into a minefield before it can visually detect that it is in a minefield. (only used for unburied mines)

Routines Called

DROP.PLOW (this appendix)

MINE.SCHED (this appendix)

PLW.ALIVE (this appendix)

WITHDRAW (this appendix)

Set

PLT.UNIT (1-D)

This set is owned by the permanent entity PLATOON LEADER. It contains the list of temporary entities owned by this platoon.

Temporary Attribute Integer

ALIVE.DEAD

Indicates whether the entity is alive or dead.

0	alive
1	dead
2	alive mounted in carrier.

FLD.AKT

Code describing the kind of action for pending internal actions.

FLD.FORM

This unit attribute gives the entity the ability to change formations when encountering an obstacle.

FLD.NO

Name of the field involved in any pending internal action.

MV.STATE

The primary control variable for initiating and stopping movement.

0	In position, do not move.
1	First call to move, do a route select and start to move.
2	Continue movement along a previously selected route.
3	Stop along the route.
4	Next position has been reached, so stop.
5	Final position has been reached, never move again.

NAM.FLD

This is the sequential id number in order of field creation.

NUM.POINTER

This variable is an argument for event QUIK.MOVE. It carries the number of the

entity to be moved.

PLT

The number of the platoon to which the entity belongs.

REP.NUMBER

This variable is an argument for event QUIK.MOVE. It carries the number of iterations of scheduling left for this event to perform.

ROUTE

Indicates the number of the route along which the element is travelling.

0 Not using a route

SYS.TYPE

This represents the general class of the system of the entity.

1 Tanks
2 mounted infantry
3 dismounted infantry
4 Artillery
5 Air
6 Air defense
7 bunker
8 Comm/ew/acq/intel
9 other

TBUMP.NUM

This unit attribute allows the entity to move in a very detailed manner as a member of his platoon when in an obstacle field.

TYP.FLD

Field Type Code.

1 - a mandatory dismount field
2 - a minefield
3 - a minefield decision field
4 - a tank ditch
5 - a road crater

6 - a blown bridge (short-wet gap)
7 - a gap decision field

Temporary Attribute Real

FLD.INT.DIST

This attribute of an entity contains the distance to a pending field internal action.

FSPEED.FAC

This unit attribute gives the entity the ability ability to slow down because of obstacle encounter.

INT.TIME

This variable is an argument for event QUIK.MOVE. It carries the time interval number when the next event is to be performed.

P1.FLD

The activation level of the minefield.

0 Not activated
1 Activated patterned minefield
# of mines Activated scatterable minefield

P3.FLD (Appendix D)

P6.FLD (Appendix D)

P7.FLD

This minefield attribute contains the lane formation number.

P8.FLD

Lane speed factor through a minefield.

P10.FLD

Plow speed factor while activated in a minefield.

P12.FLD

The minefield detection distance.

SAMAJ.FLD

The semi-major axis length of the elliptical field.

SPD

The entity's speed at the end of the most recent movement update.

Brief Explanation

Lines 18-21 Tests to determine if this minefield is activated yet. If it is not, the entity is allowed free access through the minefield.

Lines 22-30 Updates the entire platoon's location, so that the effects of a Anti-personnel frag mine can can be simulated. The distance to a mine encounter is also attained.

Lines 31-35 Checks to see if the bull through is an appropriate tactic and transfers the logic flow to the no knowledge

section of the Routine.

Lines 36-57 The entity takes the actions appropriate when he does not know the minefield exists.

Lines 38-42 If the increment is larger than one, this means that there exists a partial lane. The entity is given this safe distance to travel.

Line 43 A distance to mine encounter is obtained.

Lines 44-54 Determines if the entity is going to hit the next mine or visually detect it first.

Lines 48-52 Keeps the bull through as a separate tactic and does not allow the knowledge of the obstacle to effect the entity's movement.

Lines 58-86 Performs the actions for an entity when he knows the minefield exists.

Lines 59-64 If this route is a lane through the minefield, The formation of the entity is changed to a column, its speed is degraded, and it is allowed

to move through the minefield without detonating mines.

Line 65 — Checks on the status of the platoon's plows.

Lines 66-86 — Determines if the platoon is going to breach the minefield or to go back to the decision ellipse in order to bypass the obstacle.

Lines 67-78 — Allows the entity to go safely over the partially cleared lane. The entity's speed is degraded and his formation is changed to column.

Lines 74-77 — The speed degradation depends on the plowing status of the platoon.

Lines 80-84 — This entity's plow is being activated and the platoon is getting in the breach formation.

CODE

```
 1  ROUTINE MF.ENTRY GIVEN A,FLD
 2  DEFINE A,FLD,KNOW.LEVEL,BUMP.NUM,OBS.NUM,STAT.PLOW,
 3    INCREMENT,COUNTER,LEG AS INTEGER VARIABLES
 4  DEFINE CUM.DIST.CLEAR,MF.DET.DIST,INTERV
          AS REAL VARIABLES
 5  SUBSTITUTE THESE 4 LINES FOR MATRIX.UPDATE
 6   LET TRAF.CONTROL(OBS.NUM,ROUTE(A),1) = ROUTE(A)
 7   LET TRAF.CONTROL(OBS.NUM,ROUTE(A),3) =
                        CUM.DIST.CLEAR+FLD.INT.DIST(A)
 8   LET TRAF.CONTROL(OBS.NUM,ROUTE(A),4) =
                                    FLD.INT.DIST(A)
 9   LET TRAF.CONTROL(OBS.NUM,ROUTE(A),5) = BUMP.NUM+1
10  LET FLD.NO(A) = FLD
```

```
11  LET FLD.AKT(A) = TYP.FLD(FLD)
12  LET KNOW.LEVEL = INT.F(P6.FLD(FLD))
13  LET OBS.NUM = NAM.FLD(FLD)
14  LET INTERV = 5.0
15  LET BUMP.NUM =
                   INT.F(TRAF.CONTROL(OBS.NUM,ROUTE(A),5))
16  LET CUM.DIST.CLEAR =
                        TRAF.CONTROL(OBS.NUM,ROUTE(A),3)
17  LET INCREMENT = BUMP.NUM - TBUMP.NUM(A)
18  IF P1.FLD(FLD) EQ 0.0  ''NOT ACTIVATED YET
19     LET FLD.INT.DIST(A) = RINF.C
20     RETURN
21  OTHERWISE
22  IF SYS.TYPE(A) = 3 AND P3.FLD(FLD) GE 6.0
23    LET COUNTER =
              INT.F((SAMAJ.FLD(FLD)*2.0)/SPD(A)/INTERV)-2
24    FOR EVERY LEG IN PLT.UNIT(PLT(A)) WITH
25        ALIVE.DEAD(LEG) NE 2 AND SYS.TYPE(LEG) EQ 3
26       SCHEDULE A QUIK.MOVE GIVEN LEG,1,INTERV NOW
27    SCHEDULE A QUIK.MOVE GIVEN A,COUNTER,INTERV IN
                                        (2.0*INTERV) UNITS
28    CALL MINE.SCHED GIVEN A,FLD
29    RETURN
30  OTHERWISE
31  IF TRAF.CONTROL(OBS.NUM,ROUTE(A),2) = 0.0 OR
32        TRAF.CONTROL(OBS.NUM,ROUTE(A),2)  = 1111.0 OR
33           TRAF.CONTROL(OBS.NUM,ROUTE(A),2) = 5555.0
34     GO TO 'BULL.THRU'
35  OTHERWISE
36  IF KNOW.LEVEL LE 1  ''NO KNOWLEDGE EXISTS
37  'BULL.THRU'
38     IF INCREMENT GT 0
39        LET FLD.INT.DIST(A) = CUM.DIST.CLEAR
40        LET TBUMP.NUM(A) = BUMP.NUM
41        RETURN
42     OTHERWISE
43        CALL MINE.SCHED GIVEN A,FLD
44        LET MF.DET.DIST = P12.FLD(FLD)
45        IF MF.DET.DIST GT 0
46          IF FLD.INT.DIST(A) GT MF.DET.DIST
47            LET FLD.INT.DIST(A) = MF.DET.DIST
48            IF TRAF.CONTROL(OBS.NUM,ROUTE(A),2) = 0.0
49              LET TRAF.CONTROL(OBS.NUM,ROUTE(A),2) =
                                                   1111.0
50            ELSE
51              LET P6.FLD(FLD) = 1
52            ALWAYS
53          ALWAYS
54        ALWAYS
55        MATRIX.UPDATE
56        LET TBUMP.NUM(A) = BUMP.NUM+1
57        RETURN
58  OTHERWISE '' THEY KNOW THE MINE FIELD EXISTS
59  IF TRAF.CONTROL(OBS.NUM,ROUTE(A),3) EQ RINF.C
60     LET FLD.INT.DIST(A) = RINF.C
61     LET FLD.FORM(A) = INT.F(P7.FLD(FLD))
62     LET PSPEED.FAC(A) = P8.FLD(FLD)
63     RETURN
64  OTHERWISE
65  CALL PLW.ALIVE GIVEN A YIELDING STAT.PLOW
66  IF STAT.PLOW EQ 1
67     IF INCREMENT GT 0
68        LET FLD.INT.DIST(A) = CUM.DIST.CLEAR
69        LET TBUMP.NUM(A) = BUMP.NUM
70        IF PLT.COND(PLT(A)) = 0
71           LET PLT.COND(PLT(A)) = 1
72        ALWAYS
73        LET FLD.FORM(A) = INT.F(P7.FLD(FLD))
```

```
74          IF PLT.COND(PLT(A)) = 2
75             LET FSPEED.FAC(A) = P10.FLD(FLD)
76          ELSE
77             LET FSPEED.FAC(A) = P8.FLD(FLD)
78          ALWAYS
79          RETURN
80       OTHERWISE     ''DROP/ACTIVATE THE PLOW
81          CALL DROP.PLOW GIVEN A,FLD
82          LET TBUMP.NUM(A) = BUMP.NUM+1
83          MATRIX.UPDATE
84          RETURN
85    OTHERWISE ''GO BACKWARDS TOWARD DECISION FIELD
86    CALL WITHDRAW GIVEN A
87    RETURN
88    END
```

6. Routine DROP.PLOW

Purpose

Routine DROP.PLOW simulates the activation of mine plows for the appropriate platoon in a minefield. This routine is called from routines MF.ENTRY and MF.INTERNAL.

Given Arguments Integer

B - Pointer to the entity that is dropping its plow.

FLD- Pointer to the field in question.

Events Scheduled

HALT (this appendix)

STAND.TO (this appendix)

Permanent Attribute Integer

PLT.COND (1-D)

This attibute of the PLATOON LEADER keeps track of the present action status of the platoon in a minefield.(i.e. plowing, offsetting, pushing, etc.)

Recursive Variable Integer

TNK

Pointer to an element in the platoon of the activating entity.

Routines Called

MINE.SCHED (this appendix)

Set

PLT.UNIT (1-D)

This set is owned by the permanent entity platoon leader. It contains the list of temporary entities owned by this platoon.

Temporary Attribute Integer

ALIVE.DEAD

Indicates whether the entity is alive or dead.

0	Alive
1	Dead
2	Alive mounted in carrier.

BAGGED.BOY

This variable is an argument for event STAND.TO. It carries the number of the entity to be moved.

FLD.FORM

This unit attribute gives the entity the ability to change formations when encountering an obstacle.

FLD.NO

Name of the field involved in any pending internal action.

PLT

The number of the platoon to which the entity belongs.

SLOW.DOWN

This variable is an argument for event HALT. It carries the number of the entity to be stopped.

Temporary Attribute Real

FSPEED.FAC

This unit attribute gives the entity the ability to slow down because of obstacle encounter.

P7.FLD

This minefield attribute contains the lane formation number.

P9.FLD

This minefield attribute contains the plow activation delay time.

P10.FLD

This minefield attribute contains the plow speed factor.

Brief Explanation

Lines 3-13 Stops the platoon for a certain period of time, degrades their speed in order to plow, and schedules a start up some time in the future.

Lines 7-9 Only degrades the speed for vehicles in the minefield.

Lines 11-13 Changes the activating entity's formation to a breach column and obtains a distance to next mine encounter.

CODE

```
 1  ROUTINE DROP.PLOW GIVEN B,FLD
 2  DEFINE B,FLD,TNK AS INTEGER VARIABLES
 3         FOR EVERY TNK IN PLT.UNIT(PLT(B))
 4                         WITH ALIVE.DEAD(TNK) NE 2, DO
 5            SCHEDULE A HALT GIVEN TNK NOW
 6            SCHEDULE A STAND.TO GIVEN TNK IN
                                          P9.FLD(FLD) UNITS
 7            IF FLD.NO(TNK) EQ FLD
 8               LET FSPEED.FAC(TNK) = P10.FLD(FLD)
 9            ALWAYS
10         LOOP
11         LET PLT.COND(PLT(B)) = 2
12         LET FLD.FORM(B) = INT.F(P7.FLD(FLD))
13         CALL MINE.SCHED GIVEN B, FLD
14  RETURN
15  END
```

7. Routine WITHDRAW

Purpose

Routine WITHDRAW stops, turns around, and starts the platoon up again. This routine is called from routines MF.ENTRY and MF.INTERNAL.

Given Arguments Integer

B - Pointer to the entity that is withdrawing.

Events Scheduled

HALT (this appendix)

STAND.TO (this appendix)

TURN.AROUND (this appendix)

Recursive Variable Integer

TNK

Pointer to an element in the platoon of the activating entity.

Set

PLT.UNIT (1-D)

This set is owned by the permanent entity platoon leader. It contains the list of temporary entities owned by this platoon.

Temporary Attribute Integer

ALIVE.DEAD

Indicates whether the entity is alive or dead.

0 Alive

1 Dead
2 Alive mounted in carrier.

BAGGED.BOY

This variable is an argument for event STAND.TO. It carries the number of the entity to be moved.

DIRC.ON.RT

Indicates whether or not a vehicle is moving forward or backward on his route.

0 Vehicle is moving in order of increasing MCP numbers along the route. (forward)
1 Vehicle is moving in order of decreasing MCP numbers along the route. (backward)

END.AREA

The number which identifies the ending movement area for this entity.

PLT

The number of the platoon to which the entity belongs.

SLOW.DOWN

This variable is an argument for event HALT. It carries the number of the entity to be stopped.

START.AREA

The number which identifies the starting movement area for this entity.

TWIST

This variable is an argument for event TURN.AROUND. It carries the number of the entity to be turned In the opposite direction.

Temporary Attribute Real

FLD.INT.DIST

This attribute of an entity contains the distance to a pending field internal action.

Brief Explanation

Lines 3-11 Turns around the vehicles in the platoon that are moving forward.

Lines 6-8 Stops, turns around, and starts the appropriate entity in the next five seconds.

CODE

```
 1  ROUTINE WITHDRAW GIVEN B
 2  DEFINE B,TNK AS INTEGER VARIABLES
 3  IF DIRC.ON.RT(B) EQ 0
 4     FOR EVERY TNK IN PLT.UNIT(PLT(B))
 5                      WITH ALIVE.DEAD(TNK) NE 2, DO
 6         SCHEDULE A HALT GIVEN TNK NOW
 7         SCHEDULE A TURN.AROUND GIVEN TNK IN 4.5 UNITS
 8         SCHEDULE A STAND.TO GIVEN TNK IN 5 UNITS
 9         LET FLD.INT.DIST(TNK) = RINF.C
10     LOOP
11  ALWAYS
12  RETURN
13  END
```

8. Routine MF.INTERNAL

Purpose

Routine MF.INTERNAL allows an entity to take the appropriate internal actions in a minefield. This routine is called from routine FLD.ACT.

Given Arguments Integer

A

Pointer to the entity that activated the minefield internal action.

Global Variable Real

TRAF.CONTROL (3-D) (Appendix A)

Permanent Attribute Integer

PLT.COND (1-D)

This attibute of the PLATOON LEADER keeps track of the present action status of the platoon in a minefield.(i.e. plowing, offsetting, pushing, etc.)

Recursive Variable Integer

BUMP.NUM

Receives the number of move increments that have been made clearing the appropriate route in question. Its value is stored in the TRAF.CONTROL array.

FLD- Pointer to the field in question.

INCREMENT

Contains the number of move increments that this entity has to make to keep up with the lead element of his platoon. This number serves as a flag to determine if there is remaining cleared movement to our front or if routine MINE.SCHED has to be called.

KNOW.LEVEL

Contains the value of the field attribute P6.FLD. This is the state of intelligence known about the obstacle.

0 No knowledge exists
2 knowledge exists
3 knowledge exists and a lane exists

OBS.NUM

Receives the value of the field attribute NAM.FLD. This is the sequential id number in order of field creation.

STAT.PLOW

This variable gives the platoon's status of mine plows.

0-no plow available
1-plow available but not in use
2-plow being used
3-platoon is offsetting
4-platoon is pushing a dead plow tank

TMP.FORM

Contains the highest formation number attained by the entity's platoon.

TNK

Pointer to an element in the platoon of the activating entity.

Recursive Variable Real

CUM.DIST.CLEAR

This is the cumulative distance clear on a route through an obstacle. It receives its value from the TRAF.CONTROL array.

DRAW

Contains the value of a random number from a Uniform(0,1) distribution.

MOV.DIST

The distance that routine MOMENTUM returns. This is the distance to the lead tank in the platoon.

Routines Called

DROP.PLOW (this appendix)

HIDER

A routine used to determine the micro-terrain elevation for a selected element.

MINE.SCHED (this appendix)

MOMENTUM (this appendix)

PLW.ALIVE (this appendix)

POP.A.MINE (this appendix)

WITHDRAW (this appendix)

Set

PLT.UNIT (1-D)

This set is owned by the permanent entity PLATOON LEADER. It contains the list of temporary entities owned by this platoon.

Temporary Attribute Integer

DEFNUM

The current position or activity of an element.

1	full defilade
2	turret defilade
3	firing defilade
4	half vehicle defilade
5	moving (defilade determined by the terrain model)
6	reached final area in movement

FLD.FORM

This unit attribute gives the entity the ability to change formations when encountering an obstacle.

FLD.NO

Name of the field involved in any pending internal action.

KKILL

Indicates whether an entity has sustained a catastrophic kill.

0	No
1	yes

MKILL

Indicates whether an entity has sustained a mobility kill.

0	No
1	yes

MV.STATE

The primary control variable for initiating and stopping movement.

0	In position, do not move.
1	First call to move, do a route select and start to move,
2	Continue movement along a previously selected route.
3	Stop along the route.
4	Next position has been reached, so stop.
5	Final position has been reached, never move again.

NAM.FLD

This is the sequential id number in order of field creation.

PLOW.COND

This unit attribute gives the tank the ability to have a mine plow characteristic.

0-no plow available
1-plow available but not in use
2-plow being used
3-lead vehicle in offset
4-tank that is pushing dead plow tank

PLT

The number of the platoon to which the entity belongs.

ROUTE

Indicates the number of the route along which the element is travelling.

0	Not using a route

SYS.TYPE

This represents the general class of the system of the entity.

1	tanks
2	mounted infantry
3	dismounted infantry

4 artillery
5 air
6 air defense
7 bunker
8 comm/ew/acq/intel
9 other

TBUMP.NUM

This unit attribute allows the entity to move in a very detailed manner as a member of his platoon when in an obstacle field.

Temporary Attribute Real

FLD.INT.DIST

This attribute of an entity contains the distance to a pending field internal action.

FSPEED.FAC

This unit attribute gives the entity the ability to slow down because of an obstacle encounter.

P3.FLD

Mine Type (1-11)

1 M70 Scatterable Anti-tank mine
2 M56 Scatterable Anti-tank mine
3 M21 Hand-emplaced Anti-tank mine
4 M15 Hand-emplaced Anti-tank mine
5 M19 Hand-emplaced Anti-tank mine
6 M14 Anti-personnel blast mine
7 M25 Anti-personnel blast mine
8 M16 Anti-personnel frag mine
9 ADAM Anti-personnel frag mine
10 M74 Anti-personnel frag mine
11 Claymore Anti-personnel frag mine

P5.FLD

Probability of pushing a dead plow tank.
(1-P5.FLD) = offset probability

P6.FLD

Knowledge level of this minefield.

0	No knowledge exists.
1	No knowledge exists, the entity will see the minefield before a mine goes off.
2	knowledge exists
3	Knowledge exists and a lane exists.

P7.FLD

This minefield attribute contains the lane formation number.

P11.FLD

Pushing the dead plow speed factor.

Brief Explanation

Lines 15-20 The dismounted infantry entity detonates a mine and if he survives, a new distance to the next mine detonation is obtained.

Lines 22-30 Changes the knowledge flag so that the "bull through" option will be played.

Lines 31-48 Simulates the no knowledge exists state of this minefield.

Lines 32-34 Detonates a mine for the unaware entity.

Lines 35-37	Only changes the knowledge state when it needs to be changed. This also serves as a check to make sure the "bull through" entity does not change the knowledge state.
Lines 39-47	If the platoon has a plow it will activate it, otherwise the platoon will turn around and go back to the decision ellipse to bypass around this minefield.
Lines 49-137	Performs the appropriate internal actions when the minefield is known to exist.
Lines 50-53	If there is now a lane on this route, the rest of the platoon elements move free, undamaged.
Lines 54-69	If there is any cleared distance in front of an entity, this distance is travelled first.
Line 71	Checks the status of the platoon plows.

Lines 72-82	Performs certain actions when a platoon has an alive plow.
Lines 73-76	If the platoon is already plowing, this lead plow tank will receive a distance to the next minefield internal action (mine encounter), and will continue to plow.
Lines 77-81	If the platoon has not been plowing but has a plow or plows alive, they will now be activated.
Lines 83-86	The platoon had an alive plow on entry of the minefield, but it is now dead. The platoon will now withdraw and take a bypass around the this obstacle.
Lines 87-89	If the "bull through" tactic is appropriate, the platoon will begin on an offset sequence.
Lines 90-94	If the platoon has offset or pushed the dead plow tank before and the pusher tank is dead (if applicable), then an offset sequence is started or continued.

Lines 95-134 If the platoon had been plowing and the plow tanks are dead, the platoon has the option to either offset around the dead plow or push The dead plow tank.

Lines 97-102 This entity will push the dead plow tank.

Lines 103-108 Changes the formation of the entire platoon in this minefield for a first time offset.

Lines 109-114 Upgrades the speed of the entity to the maximum Allowed by the terrain and it reassures that This entity is the lead element of his platoon. A mine detonation also occurs.

Lines 115-132 Offsets the rest of the platoon elements if the activating entity is at least a mobility kill.

Lines 129-131 If this is a patterned minefield and a tank has died in this mine belt, the other tanks in the activating entity's platoon must still go through this same belt.

Lines 135-137 Obtain a new mine encounter distance and update the TRAF.CONTROL array.

CODE

```
 1  ROUTINE MF.INTERNAL GIVEN A
 2  DEFINE A,FLD,KNOW.LEVEL,BUMP.NUM,OBS.NUM,STAT.PLOW,
 3         INCREMENT,TNK, TMP.FORM AS INTEGER VARIABLES
 4  DEFINE CUM.DIST.CLEAR,DRAW,MOV.DIST
          AS REAL VARIABLES
 5  SUBSTITUTE THESE 3 LINES FOR MATRIX.UPDATE
 6   LET TRAF.CONTROL(OBS.NUM,ROUTE(A),3) =
                          CUM.DIST.CLEAR+FLD.INT.DIST(A)
 7   LET TRAF.CONTROL(OBS.NUM,ROUTE(A),4) =
                                    FLD.INT.DIST(A)
 8   LET TRAF.CONTROL(OBS.NUM,ROUTE(A),5) = BUMP.NUM+1
 9  LET FLD = FLD.NO(A)
10  LET KNOW.LEVEL = INT.F(P6.FLD(FLD))
11  LET OBS.NUM = NAM.FLD(FLD)
12  LET BUMP.NUM=INT.F(TRAF.CONTROL(OBS.NUM,ROUTE(A),5))
13  LET CUM.DIST.CLEAR= TRAF.CONTROL(OBS.NUM,ROUTE(A),3)
14  LET INCREMENT = BUMP.NUM - TBUMP.NUM(A)
15  IF SYS.TYPE(A) = 3 ''DISMOUNTED INFANTRY
16     CALL POP.A.MINE GIVEN A
17     IF KKILL(A) NE 1  ''A SOLDIER HIT IS DEAD
18        CALL MINE.SCHED GIVEN A, FLD
19     ALWAYS
20     RETURN
21  OTHERWISE
22  IF TRAF.CONTROL(OBS.NUM,ROUTE(A),2) = 0.0 OR
23        TRAF.CONTROL(OBS.NUM,ROUTE(A),2) = 1111.0
24     IF TRAF.CONTROL(OBS.NUM,ROUTE(A),2) = 0.0
25        LET KNOW.LEVEL = 0
26     ELSE
27        LET KNOW.LEVEL = 1
28     ALWAYS
29     LET TRAF.CONTROL(OBS.NUM,ROUTE(A),2) = 5555.0
30  ALWAYS
31  IF KNOW.LEVEL LE 1  ''NO KNOWLEDGE EXISTS
32     IF KNOW.LEVEL EQ 0
33        CALL POP.A.MINE(A)
34     ALWAYS
35     IF P6.FLD(FLD) LT 2
36        LET P6.FLD(FLD) = 2  '' KNOWLEDGE EXISTS
37     ALWAYS
38     IF TRAF.CONTROL(OBS.NUM,ROUTE(A),2) NE 5555.0
          ''DROP PLOW OR GO BACK
39        CALL PLW.ALIVE GIVEN A YIELDING STAT.PLOW
40        IF STAT.PLOW EQ 1  ''DROP/ACTIVATE THE PLOW
41           CALL DROP.PLOW GIVEN A,FLD
42           LET TBUMP.NUM(A) = BUMP.NUM+1
43           MATRIX.UPDATE
44           RETURN
45        OTHERWISE ''GO BACKWARDS TOWARD DECISION FIELD
46        CALL WITHDRAW GIVEN A
47        RETURN
48     OTHERWISE  ''BULL-THRU
49  ALWAYS ''KNOWLEDGE EXISTS ABOUT THE MINEFIELD
50     IF TRAF.CONTROL(OBS.NUM,ROUTE(A),3) EQ RINF.C
51        LET FLD.INT.DIST(A) =
                          TRAF.CONTROL(OBS.NUM,ROUTE(A),3)
52        RETURN
53     ALWAYS
```

```
54      IF INCREMENT GT 0
55         IF INCREMENT EQ 1
56            LET FLD.INT.DIST(A) =
                        TRAF.CONTROL(OBS.NUM,ROUTE(A),4)
57         ELSE
58            CALL MOMENTUM GIVEN A,BUMP.NUM
                                          YIELDING MOV.DIST
59            LET FLD.INT.DIST(A) = MOV.DIST
60         ALWAYS
61         LET TBUMP.NUM(A) = BUMP.NUM
62         FOR EVERY TNK IN PLT.UNIT(PLT(A))
63                        WITH SYS.TYPE(TNK) NE 3, DO
64             IF FLD.FORM(TNK) GT TMP.FORM
65                LET TMP.FORM = FLD.FORM(TNK)
66             ALWAYS
67         LOOP
68         LET FLD.FORM(A) = TMP.FORM
69         RETURN
70      OTHERWISE
71         CALL PLW.ALIVE GIVEN A YIELDING STAT.PLOW
72         IF STAT.PLOW EQ 1
73            IF PLT.COND(PLT(A)) EQ 2
74               CALL MINE.SCHED GIVEN A, FLD
75               LET TBUMP.NUM(A) = BUMP.NUM+1
76            ELSE  ''TO DROP PLOW
77               CALL DROP.PLOW GIVEN A,FLD
78               LET TBUMP.NUM(A) = BUMP.NUM+1
79            ALWAYS
80            MATRIX.UPDATE
81            RETURN
82         OTHERWISE        ''THE PLOWS ARE DEAD
83         IF PLT.COND(PLT(A)) EQ 1
              ''GO BACKWARDS TOWARD DECISION FIELD
84            CALL WITHDRAW GIVEN A
85            RETURN
86         OTHERWISE
87         IF PLT.COND(PLT(A)) EQ 0  ''BULL-THRU CASE
88            GO TO 'OFFSET'
89         OTHERWISE
90         IF PLT.COND(PLT(A)) GE 3
              ''PUSHED OR OFFSET BEFORE
91            IF STAT.PLOW LE 3
                 ''PUSHER OR LEAD TANK DEAD
92               GO TO 'OFFSET'
93            OTHERWISE
94         ALWAYS  ''PUSHER OR LEAD TANK STILL ALIVE
95         IF PLT.COND(PLT(A)) EQ 2
              ''HAD BEEN PLOWING BUT PLOW DIED
96            LET DRAW = UNIFORM.F(0.,1.,7)
97            IF DRAW LE P5.FLD(FLD)
                 ''PUSH THE DEAD PLOW TANK
98               LET FSPEED.FAC(A) = P11.FLD(FLD)
99               LET PLOW.COND(A) = 4
100              LET DEFNUM(A) = 4 ''1/2''  CALL HIDER(A)
101              LET PLT.COND(PLT(A)) = 4
102           ELSE  ''OFFSET AROUND DEAD PLOW TANK
103              FOR EVERY TNK IN PLT.UNIT(PLT(A))
104                           WITH SYS.TYPE(TNK) NE 3, DO
105                 IF FLD.NO(TNK) EQ FLD
106                    LET FLD.FORM(TNK) =
                                    INT.F(P7.FLD(FLD))+1
107                 ALWAYS
108              LOOP
109  'OFFSET'    LET FSPEED.FAC(A) = 1.0
110              LET PLOW.COND(A) = 3
111              IF PLT.COND(PLT(A)) NE 0
                    ''NOT BULL-THRU CASE
112                 CALL POP.A.MINE(A)
```

```
113                 ALWAYS
114                 LET PLT.COND(PLT(A)) = 3
115                 IF KKILL(A) = 1 OR MKILL(A) = 1
116                   FOR EVERY TNK IN PLT.UNIT(PLT(A))
117                           WITH SYS.TYPE(TNK) NE 3, DO
118                   IF KKILL(TNK) NE 1 AND MKILL(TNK) NE 1
119                           IF FLD.NO(TNK) EQ FLD
120                             IF FLD.FORM(TNK) LT
121                                          INT.F(P7.FLD(FLD))
                                   ''BULL-THRU OFFSET
122                                LET FLD.FORM(TNK) =
                                             INT.F(P7.FLD(FLD))+1
123                             ELSE''REGULAR OFFSET
124                                ADD 1 TO FLD.FORM(TNK)
125                             ALWAYS
126                         ALWAYS
127                   ALWAYS
128                   LOOP
129                    IF P3.FLD(FLD) GE 3 AND
                                          P3.FLD(FLD) LT 6
130                      RETURN
131                    ALWAYS
132                 ALWAYS
133               ALWAYS
134         ALWAYS
135         CALL MINE.SCHED GIVEN A, FLD
136         MATRIX.UPDATE
137         LET TBUMP.NUM(A) = BUMP.NUM+1
138   RETURN
139   END
```

9. Routine MF.EXIT

Purpose

Routine MF.EXIT performs the minefield exit actions for the activating/moving entity. This routine is called from routine FLD.ACT.

Given Arguments Integer

A

Pointer to the entity that activated the minefield internal action.

Global Variable Real

TRAF.CONTROL (3-D) (Appendix A)

Permanent Attribute Integer

PLT.COND (1-D)

This attibute of the PLATOON LEADER keeps track of the present action status of the platoon in a minefield. (i.e. plowing, offsetting, pushing, etc.)

Recursive Variable Integer

FLD- Pointer to the field in question.

OBS.NUM

Receives the value of the field attribute NAM.FLD. This is the sequential id number in order of field creation.

Routines Called

HIDER

A Routine used to determine the micro-terrain elevation for a selected element.

Temporary Attribute Integer

DEFNUM

The current position or activity of an element.

1	full defilade
2	turret defilade
3	firing defilade
4	half vehicle defilade
5	moving (defilade determined by the terrain model)
6	Reached final area in movement

DIRC.ON.RT

Indicates whether or not a vehicle is moving forward or backward on his route.

0	Vehicle is moving in order of increasing MCP numbers along the route. (forward)
1	Vehicle is moving in order of decreasing MCP numbers along the route. (backward)

FLD.FORM

This unit attribute gives the entity the ability to change formations when encountering an obstacle.

FLD.NO

Name of the field involved in any pending internal action.

NAM.FLD

This is the sequential id number in order of field creation.

PLOW.COND

This unit attribute gives the tank the ability to have a mine plow characteristic.

0-no plow available
1-plow available but not in use
2-plow being used
3-lead vehicle in offset
4-tank that is pushing dead plow tank

PLT

The number of the platoon to which the entity belongs.

ROUTE

Indicates the number of the route along which the element is travelling.

0 Not using a route

SYS.TYPE

This represents the general class of the system of the entity.

1 tanks
2 mounted infantry
3 dismounted infantry
4 artillery
5 air
6 air defense
7 bunker
8 comm/ew/acq/intel
9 other

TBUMP.NUM

This unit attribute allows the entity to move in a very detailed manner as a member of his platoon when in an obstacle field.

Temporary Attribute Real

FSPEED.FAC

This unit attribute gives the entity the ability to slow down because of obstacle encounter.

P1.FLD

The activation level of the minefield.

0	Not activated
1	Activated patterned minefield
# of mines	Activated scatterable minefield

P6.FLD

knowledge level of this minefield.

0	no knowledge exists
2	knowledge exists
3	knowledge exists and a lane exists.

Brief Explanation

Lines 3-5 If the entity is dismounted infantry, do nothing.

Lines 9-18 Updates the TRAF.CONTROL array for this cleared lane and lets others know that a lane exists on this route through the minefield.

Lines 10-12 Does not let the knowledge level change if the entity has not hit or seen a mine.

Lines 13-15 Prevents TRAF.CONTROL array changes when the minefield is not activated yet.

Lines 16-19 If the entity was the push tank or the lead element, then this status of the entity no longer holds.

Lines 21-23 Re-initializes changed entity attributes.

CODE

```
 1  ROUTINE MF.EXIT GIVEN A
 2  DEFINE A,FLD,OBS.NUM AS INTEGER VARIABLES
 3  IF SYS.TYPE(A) = 3 ''DISMOUNTED INFANTRY
 4     RETURN
 5  OTHERWISE
 6  LET FLD = FLD.NO(A)
 7  LET OBS.NUM = NAM.FLD(FLD)
 8  LET FSPEED.FAC(A) = 1.0
 9  IF DIRC.ON.RT(A) = 0
10     IF TBUMP.NUM(A) GT 1
11        LET P6.FLD(FLD) = 3
12     ALWAYS
13     IF P1.FLD(FLD) NE 0.0  ''ACTIVE MINEFIELD
14        LET TRAF.CONTROL(OBS.NUM,ROUTE(A),3) = RINF.C
15     ALWAYS
16     IF PLOW.COND(A) GT 1
17        LET DEFNUM(A) = 5 ''MOVING'' CALL HIDER(A)
18        LET PLOW.COND(A) = 0
19     ALWAYS
20  ALWAYS
21     LET FLD.FORM(A) = 0
22     LET TBUMP.NUM(A) = 0
23     LET PLT.COND(PLT(A)) = 0
24  RETURN
25  END
```

10. Event HALT

Purpose

Event HALT stops the moving entity. This event is important because it allows the user to get out of the FLD.ACT and MOVE routines. This event is scheduled from routines DROP.PLOW, WITHDRAW, GAP.ENTRY, and GAP.INTERNAL.

Given Arguments Integer

B - The pointer to the entity that has activated this stop event.

Temporary Attribute Integer

MV.STATE

The primary control variable for initiating and stopping movement.

0	In position, do not move.
1	First call to move, do a route select and start to move.
2	Continue movement along a previously selected route.
3	Stop along the route.
4	Next position has been reached, so stop.
5	Final position has been reached, never move again.

SLOW.DOWN

This variable is an argument for event HALT. It carries the number of the entity to be stopped.

Brief Explanation

Lines 1-5 Changes the MV.STATE of the entity so that it will slow down and stop moving.

CODE

```
1  EVENT HALT GIVEN B
2  DEFINE B AS AN INTEGER VARIABLE
3  LET MV.STATE(B) = 3
4  RETURN
5  END
```

11. Event TURN.AROUND

Purpose

Event TURN.AROUND turns the entity around in the opposite direction of present travel. This event is scheduled from Routines MF.DECISION, WITHDRAW, GAP.DECISION, GAP.ENTRY, and from event HEAVY.JUNK.

Given Arguments Integer

B - The pointer to the entity that is going to turn around.

Recursive Variable Integer

TMP.AREA

This variable stores the value of the STATE.AREA, so that the values of START.AREA and END.AREA can be switched.

Temporary Attribute Integer

END.AREA

The number which identifies the ending movement area for this entity.

KKILL

Indicates whether an entity has sustained a catastrophic kill.

0 No
1 yes

MKILL

Indicates whether an entity has sustained a mobility kill.

0 No
1 yes

START.AREA

The number which identifies the starting movement area for this entity.

TWIST

This variable is an argument for event TURN.AROUND. It carries the number of the entity to be turned in the opposite direction.

Brief Explanation

Lines 3-7 Switches the START.AREA and END.AREA of the alive entity in question.

CODE

```
1  EVENT TURN.AROUND GIVEN B
2  DEFINE B,TMP.AREA AS INTEGER VARIABLES
3  IF KKILL(B) NE 1 AND MKILL(B) NE 1
4    LET TMP.AREA = START.AREA(B)
5    LET START.AREA(B) = END.AREA(B)
6    LET END.AREA(B) = TMP.AREA
7  ALWAYS
8  RETURN
9  END
```

12. Event STAND.TO

Purpose

Event STAND.TO starts the activating entity moving again. This event is scheduled from routines DROP.PLOW, WITHDRAW, and GAP.INTERNAL. It is also scheduled from events GAP.BREACH, HEAVY.JUNK, and GAP.JOCK.

Given Argument Integer

B - The pointer to the entity that is going to start moving again.

Temporary Attribute Integer

BAGGED.BOY

This variable is an argument for Event STAND.TO. It carries the number of the entity to be moved.

KKILL

Indicates whether an entity has sustained a catastrophic kill.

0 no
1 yes

MKILL

Indicates whether an entity has sustained a mobility kill.

0 no
1 yes

MV.STATE

The primary control variable for initiating and stopping movement.

0 In position, do not move.

1	First call to move, do a route select and start to move.
2	Continue movement along a previously selected route.
3	Stop along the route.
4	Next position has been reached, so stop.
5	Final position has been reached, never move again.

Temporary Attribute Real

SPD

The entity's speed at the end of the most recent movement update.

T.SPD

The simulation time at which the most recent movement update ended. The time that SPD was last set.

Brief Explanation

Lines 3-7 Starts the entity moving again now from a standstill.

CODE

```
1  EVENT STAND.TO GIVEN B
2  DEFINE B AS AN INTEGER VARIABLE
3  IF KKILL(B) NE 1 AND MKILL(B) NE 1
4     LET T.SPD(B) = TIME.V
5     LET MV.STATE(B) = 2
6     LET SPD(B) = 0.0
7  ALWAYS
8  RETURN
9  END
```

13. Event DIVERT

Purpose

Event DIVERT allows/triggers lateral movement for an entity going from one decision ellipse to another. This event is scheduled from routines MF.DECISION and GAP.DECISION.

Given Arguments Integer

B - The pointer to the entity that is going to start lateral movement.

MCP

This argument carries the movement control point that will cause lateral movement.

RT

This argument carries the route number that will cause lateral movement.

Temporary Attribute Integer

BYPASS

This variable is an argument for event DIVERT. It carries the route number that will cause lateral movement.

DIRC.ON.RT

Indicates whether or not a vehicle is moving forward or backward on his route.

0 Vehicle is moving in order of increasing MCP numbers along the route. (forward)

1 Vehicle is moving in order of decreasing MCP numbers along the route. (backward)

FORMACODE

The formation number of the formation to be used by the platoon.

0 Not in formation, vehicle moves along route without offset.

HOLD.FORM

This unit attribute is a storage place, where the units formation number for the platoon (FORMACODE) can be placed when lateral route movement is appropriate.

HOME

This variable is an argument for event DIVERT. It carries the MCP number that will be cause lateral movement.

NEXT.MCP

Movement control point number (on the designated route) toward which the element is now moving.

0 end of route has been reached

ROUTE

Indicates the number of the route along which the element is travelling.

0 not using a route

SIDE.STEPPER

This variable is an argument for event DIVERT. It carries the number of the entity to be moved laterally.

Brief Explanation

Lines 3-4 Directs the entity to go to a Movement Control Point on a new route.

Lines 5-6 Enables the entity to move straight to the MCP on this new route.

Line 7 Makes sure the entity is not moving backwards (away from the attack).

CODE

```
1  EVENT DIVERT GIVEN B, RT, MCP
2  DEFINE B, RT, MCP AS INTEGER VARIABLES
3  LET ROUTE(B) = RT
4  LET NEXT.MCP(B) = MCP
5  LET HOLD.FORM(B) = FORMACODE(B)
6  LET FORMACODE(B) = 0
7  LET DIRC.ON.RT(B) = 0
8  RETURN
9  END
```

14. Routine MOMENTUM

Purpose

Routine MOMENTUM allows a platoon member to move over the remaining cleared distance on a route through a minefield. This routine is called from routine MF.INTERNAL.

Given Arguments Integer

B - The pointer to the entity that needs a distance to to travel in the minefield to keep up with his platoon.

BUMP

Carries the number of move increments that have been made clearing the appropriate route in question. Its value is taken from the TRAF.CONTROL array.

Yielding Argument Real

MOV.DIST

This is the distance to the lead tank in the platoon from the activating entity.

Recursive Variable Integer

TNK

Pointer to an element in the platoon of the activating entity.

Recursive Variable Real

MOV.LENG

This is the distance left to be cleared on a route through the minefield.

Set

PLT.UNIT (1-D)

This set is owned by the permanent entity platoon leader. It contains the list of temporary entities owned by this platoon.

Temporary Attribute Integer

PLT

The number of the platoon to which the entity belongs.

SYS.TYPE

This represents the general class of the system of the entity.

1	tanks
2	mounted infantry
3	dismounted infantry
4	artillery
5	air
6	air defense
7	bunker
8	comm/ew/acq/intel
9	other

TBUMP.NUM

This unit attribute allows the entity to move in a very detailed manner as a member of his platoon when in an obstacle field.

Temporary Attribute Real

FLD.BDY.DIST

This is the distance from the entity to the nearest field boundary.

FLD.INT.DIST

This attribute of an entity contains the distance to a pending field internal action.

Brief Explanation

Lines 4-8 Searches the platoon members to find the lead vehicle and his distance to clear the entire lane in the minefield.

Line 9 Obtains the distance that has already been cleared in front of the activating entity.

CODE

```
 1  ROUTINE MOMENTUM GIVEN B,BUMP YIELDING MOV.DIST
 2  DEFINE B,BUMP,TNK AS INTEGER VARIABLES
 3  DEFINE MOV.DIST,MOV.LENG AS A REAL VARIABLE
 4  FOR EVERY TNK IN PLT.UNIT(PLT(B)) WITH
                                  SYS.TYPE(TNK) NE 3, DO
 5      IF TBUMP.NUM(TNK) EQ BUMP
 6         LET MOV.LENG =
                      FLD.BDY.DIST(TNK)-FLD.INT.DIST(TNK)
 7      ALWAYS
 8  LOOP
 9  LET MOV.DIST = FLD.BDY.DIST(B)-MOV.LENG
10  RETURN
11  END
```

15. Routine PLW.ALIVE

Purpose

Routine PLW.ALIVE checks the status of the platoon's plows. It also checks to see if a platoon member is a push tank or the lead vehicle participating in an offset. This routine is called from routines MF.DECISION, MF.ENTRY, and MF.INTERNAL.

Given Argument Integer

B - The pointer to the entity that needs a status on the plows within his platoon.

Yielding Argument Integer

STAT.PLOW

This variable gives the platoon's status of mine plows.

0-no plow available
1-plow available but not in use
2-plow being used
3-platoon is offsetting
4-platoon is pushing a dead plow tank

Recursive Variable Integer

J - Receives the status of the platoon plows.

TNK

Pointer to an element in the platoon of the activating entity.

Set

PLT.UNIT (1-D)

This set is owned by the permanent entity platoon leader. It contains the list of temporary entities owned by this platoon.

Temporary Attribute Integer

KKILL

Indicates whether an entity has sustained a catastrophic kill.

0 No
1 yes

MKILL

Indicates whether an entity has sustained a mobility kill.

0 No
1 yes

PLOW.COND

This unit attribute gives the tank the ability to have a mine plow characteristic.

0-no plow available
1-plow available but not in use
2-plow being used
3-lead vehicle in offset
4-tank that is pushing dead plow tank

PLT

The number of the platoon to which the entity belongs.

Brief Explanation

Lines 3-20 Searches the platoon for its highest plow condition/status. The platoon

can only have one such number at a time.

CODE

```
 1   ROUTINE PLW.ALIVE GIVEN B YIELDING STAT.PLOW
 2   DEFINE B,TNK,J,STAT.PLOW AS INTEGER VARIABLES
 3   FOR EVERY TNK IN PLT.UNIT(PLT(B)), DO
 4       IF KKILL(TNK) EQ 1 OR MKILL(TNK) EQ 1
 5       CYCLE
 6       OTHERWISE
 7       LET J = PLOW.COND(TNK)
 8       IF J = 1
 9       LET STAT.PLOW = 1
10       RETURN
11       OTHERWISE
12       IF J = 3
13       LET STAT.PLOW = 3
14       RETURN
15       OTHERWISE
16       IF J = 4
17       LET STAT.PLOW = 4
18       RETURN
19       OTHERWISE
20   LOOP
21   LET STAT.PLOW = 0
22   RETURN
23   END
```

16. Event QUIK.MOVE

Purpose

Event QUIK.MOVE updates the location of an entity. This event continues to update locations for a certain number of times at a prescribed interval. Event QUIK.MOVE is scheduled from routines MF.DECISION, MF.ENTRY, and GAP.DECISION. It is also scheduled from events QUIK.MOVE, WALL.BREACH, GAP.BREACH, HEAVY.JUNK, and GAP.JOCK.

Given Arguments Integer

B - This argument carries the number of the entity to be moved.

COUNT

This argument carries the number of iterations of scheduling left for this event to perform.

Given Argument Real

INTERVAL

A constant term that is set so that element locations are updated every five seconds during a small portion of the simulation.

Routines Called

LOC

A routine used to determine whether movement is possible and to initiate a call to MOVE.

Temporary Attribute Integer

KKILL

Indicates whether an entity has sustained a catastrophic kill.

0	no
1	yes

MKILL

Indicates whether an entity has sustained a mobility kill.

0	no
1	yes

NUM.POINTER

This variable is an argument for event QUIK.MOVE. It carries the number of the entity to be moved.

REP.NUMBER

This variable is an argument for event QUIK.MOVE. It carries the number of iterations of scheduling left for this event to perform.

Temporary Attribute Real

INT.TIME

This variable is an argument for event QUIK.MOVE. It carries the time interval number when the next event is to be performed.

Brief Explanation

Lines 4-6 Updates the location of the entity, decrements the number of future schedulings, and schedules another event.

CODE

```
 1   EVENT QUIK.MOVE GIVEN B, COUNT, INTERVAL
 2   DEFINE B,COUNT AS INTEGER VARIABLES
 3   DEFINE INTERVAL AS A REAL VARIABLE
 4   CALL LOC(B)
 5   IF COUNT GT 0 AND KKILL(B) NE 1 AND MKILL(B) NE 1
 6   SCHEDULE A QUIK.MOVE GIVEN B, COUNT-1, INTERVAL IN
                                              INTERVAL UNITS
 7   ALWAYS
 8   RETURN
 9   END
```

17. Routine MINE.SCHED

Purpose

Routine MINE.SCHED produces the distance that an activating minefield entity has to travel in order to encounter a mine. This routine is called from routines DROP.PLOW, MF.ENTRY, and MF.INTERNAL.

Given Arguments Integer

FLD- Pointer to the field in question.

VEH- Pointer to the element being moved.

Global Variable Real

TARDIM (3-D)

Contains the target dimensions of all entity types in the simulation. This is indexed by SYS.TYPE and WPN.TYPE of the element.

Permanent Attribute Integer

NXT.BELT (1-D)

This PLATOON LEADER attribute stores the number of the next mine belt to be encountered in the minefield. It allows the platoon to move through a minefield as an organized unit.

Recursive Variable Integer

TNK

Pointer to an element in the platoon of the activating entity.

Recursive Variable Real

ANGLE

Set to the direction of movement of the entity.

DRAW

Contains the value of a random number from a Uniform (0,1) distribution.

EXPECT.DIST

The reciprocal of the width times the density of the scatterable minefield.

MF.DENSITY

Is the number of mines in the minefield divided by the area of the field.

THETA

Is the direction of movement of the entity minus the orientation angle of the ellipse.

WIDTH

Takes on several values during this routine depending upon the situation.

Trip wire length (TWL) - If infantry in an anti-personnel frag field.
Two times the TWL plus the VW - If a vehicle is in an anti-personnel frag field.
Vehicle width - If a vehicle is in an anti-tank field.

Set

PLT.UNIT (1-D)

This set is owned by the permanent entity PLATOON LEADER. It contains the list of temporary entities owned by this platoon.

Temporary Attribute Integer

COLOR

Indicates the color of the element.

0	red (attacker)
1	blue (defender)

FLD.NO

Name of the field involved in any pending internal action.

NAM.FLD

This is the sequential id number in order of field creation.

NAME

The element number of the entity.

PLT

The number of the platoon to which the entity belongs.

SYS.TYPE

This represents the general class of the system of the entity.

1	tanks
2	mounted infantry
3	dismounted infantry
4	artillery
5	air
6	air defense
7	bunker
8	comm/ew/acq/intel
9	other

WHAT.BELT

This entity attribute keeps the updated belt number that is to be encountered next.

WPN.TYPE

Describes the specific system within the system code. For example, system type 1 is a tank and weapon type four for this system indicates an AVLB.

Temporary Attribute Real

ANGLE.FLD

Orientation angle in radians measured counterclockwise from the east to the major axis of the elliptical field.

AREA.FLD

This field attribute carries the area of the field in meters squared.

DIR.OF.MVMT

Indicates the entity's direction of movement measured in radians from east.

FLD.INT.DIST

This attribute of an entity contains the distance to a pending field internal action.

P1.FLD

The activation level of the minefield

0	not activated
1	activated patterned minefield
# of mines	activated scatterable minefield

P2.FLD (Appendix D)

P3.FLD

Mine Type (1-11)

1	M70 Scatterable Anti-tank mine
2	M56 Scatterable Anti-tank mine
3	M21 Hand-emplaced Anti-tank mine
4	M15 Hand-emplaced Anti-tank mine
5	M19 Hand-emplaced Anti-tank mine
6	M14 Anti-personnel blast mine
7	M25 Anti-personnel blast mine
8	M16 Anti-personnel frag mine
9	ADAM Anti-personnel frag mine
10	M74 Anti-personnel frag mine
11	Claymore Anti-personnel frag mine

SAMAJ.FLD

The semi-major axis length of the elliptical field.

X.CURRENT

The X-coordinate for the entity as of the last movement update.

Y.CURRENT

The Y-coordinate for the entity as of the last movement update.

Brief Explanation

Lines 4-7 Checks if the minefield is still active. This is done primarily for the Anti-personnel minefields.

Lines 8-35 Simulates the Anti-personnel minefields.

Lines 9-19 The Anti-personnel blast minefield is modelled.

Line 10	Determines the minefield density in mines per square meter.
Lines 11-14	Calculates the distance to a mine encounter for a vehicle moving through an Anti-personnel blast minefield.
Lines 15-18	Calculates the distance to a mine encounter for a dismounted soldier moving through an Anti-personnel blast minefield.
Lines 20-35	Simulates the Anti-personnel frag minefield.
Lines 21-22	Sets the distance to a Claymore mine detonation.
Lines 24-26	Calculates the width of the lane made by a vehicle moving through an Anti-personnel frag minefield.
Line 28	Obtains the effective length of the AP frag mine in question.
Lines 30-32	Calculates the minefield density. This density and the effective lane width are used to obtain an average

mine encounter distance. The exponential distribution is then sampled to get the actual distance to the next mine.

Line 35 Decrements the number of mines in the minefield.

Lines 36-58 Models the effects of Anti-tank minefields.

Lines 38-42 Obtains a minefield density and the width of of the activating entity. An average distance to mine encounter is calculated. A distance is sampled from an exponential distribution using the above average distance value.

Lines 43-58 Simulates a patterned/belted minefield.

Lines 44-47 Updates the next mine belt to be encountered by the platoon elements.

Lines 48-54 Calculates the distance to the next mine encounter using the distance between the belts and the orientation

of the minefield. Red/Attacking elements only are affected.

Lines 55-57 Simulates that blue elements know where the planned mine lanes are in their own minefields. These minefields have no effect on the defender.

Lines 59-61 Dismounted infantry will not detonate Anti-tank mines.

CODE

```
 1  ROUTINE MINE.SCHED GIVEN VEH,FLD
 2  DEFINE VEH,FLD,TNK AS INTEGER VARIABLES
 3  DEFINE DRAW,MF.DENSITY,EXPECT.DIST,ANGLE,THETA,WIDTH
          AS REAL VARIABLES
 4  IF P1.FLD(FLD) = 0.0 ''NOT ACTIVE ANY MORE
 5     LET FLD.INT.DIST(VEH) = RINF.C
 6     RETURN
 7  OTHERWISE
 8  IF P3.FLD(FLD) GE 6 ''ANTI-PERSONNEL MINES
 9     IF P3.FLD(FLD) LT 8  '' MINE.TYPE=AP BLAST
10        LET MF.DENSITY = P1.FLD(FLD)/AREA.FLD(FLD)
          ''P1.FLD =NUMBER.MINES
11        IF SYS.TYPE(VEH) NE 3
12           LET WIDTH =
                TARDIM(SYS.TYPE(VEH),WPN.TYPE(VEH),6)
13           LET EXPECT.DIST =
                          1.0/(WIDTH*MF.DENSITY)/3.0
14           LET FLD.INT.DIST(VEH) =
                          EXPONENTIAL.F(EXPECT.DIST,7)
15        ELSE
16           LET DRAW=UNIFORM.F(0.,1.,7)
17           LET DRAW=1-DRAW
18           LET FLD.INT.DIST(VEH) =
                ((-2.5)*LOG.E.F(DRAW))/(MF.DENSITY**.12)
19        ALWAYS
20     ELSE      ''MINE TYPE = AP FRAG
21        IF P3.FLD(FLD)  EQ 11.0
             ''CLAYMORE MINE 14 BY 24 BY 90 DEGREES
22           LET FLD.INT.DIST(VEH) = .2*SAMAJ.FLD(FLD)
23        ELSE
24           IF SYS.TYPE(VEH) NE 3
25              LET WIDTH =
                   TARDIM(SYS.TYPE(VEH),WPN.TYPE(VEH),6)+
26                                        (P2.FLD(FLD)/.707)
27           ELSE
28              LET WIDTH = P2.FLD(FLD)/.707
29           ALWAYS
```

```
30            LET MF.DENSITY = P1.FLD(FLD)/AREA.FLD(FLD)
31            LET EXPECT.DIST = 1.0/(WIDTH*MF.DENSITY)
32            LET FLD.INT.DIST(VEH) =
                              EXPONENTIAL.F(EXPECT.DIST,7)
33         ALWAYS
34       ALWAYS
35       SUBTRACT 1 FROM P1.FLD(FLD)
36     ELSE ''ANTI-TANK MINES
37       IF SYS.TYPE(VEH) NE 3 ''SOME TYPE OF VEHICLE
38       IF P3.FLD(FLD) LT 3
            '' P3.FLD = MINE.TYPE // SCATTERABLE MINES
39          LET MF.DENSITY = P1.FLD(FLD)/AREA.FLD(FLD)
             ''P1.FLD =NUMBER.MINES
40          LET WIDTH =
                    TARDIM(SYS.TYPE(VEH),WPN.TYPE(VEH),6)
41          LET EXPECT.DIST = 1.0/(WIDTH*MF.DENSITY)
42          LET FLD.INT.DIST(VEH) =
                              EXPONENTIAL.F(EXPECT.DIST,7)
43       ELSE ''A BELTED MINEFIELD
44          ADD 1 TO NXT.BELT(PLT(VEH))
45          FOR EVERY TNK IN PLT.UNIT(PLT(VEH))
46                                 WITH SYS.TYPE(TNK) NE 3
47          LET WHAT.BELT(TNK) =NXT.BELT(PLT(VEH))
48          IF COLOR(VEH) = 0 ''RED FORCE WILL HIT MINES
49            IF DIR.OF.MVMT(VEH) LT 0.0
50              LET ANGLE = 360/RADIAN.C + DIR.OF.MVMT(VEH)
51            ELSE LET ANGLE = DIR.OF.MVMT(VEH)
52            ALWAYS LET THETA = ANGLE - ANGLE.FLD(FLD)
53                  ''P2.FLD = BELT.SEPARATION
54             LET FLD.INT.DIST(VEH) =
                              P2.FLD(FLD)/ABS.F(SIN.F(THETA))
55          ELSE ''BLUE FORCE KNOWS WHERE THE LANES ARE
56            LET FLD.INT.DIST(VEH) = RINF.C
57          ALWAYS
58       ALWAYS
59       ELSE ''DISMOUNTED INFANTRY IN A AT MINEFIELD
60        LET FLD.INT.DIST(VEH) = RINF.C
61       ALWAYS
62     ALWAYS
63     PRINT 1 LINE WITH NAME(VEH),X.CURRENT(VEH),
               Y.CURRENT(VEH),TIME.V,
64             NAM.FLD(FLD),FLD.INT.DIST(VEH) AS FOLLOWS
MINE DET SCHED VEH **** LOCN ****** ****** T=**** FLD=***
                        DETD=******.**
65     RETURN
66     END
```

18. Routine GAP.DECISION

Purpose

Routine GAP.DECISION allows the entity and his platoon the options of going to the gap obstacle to attempt a bull through or breach, and the option to bypass or lane select, whichever is applicable. This routine is called from routine FLD.ACT.

Given Arguments Integer

A

Pointer to the entity that has entered the decision ellipse.

FLD- pointer to the field in question.

Events Scheduled

DIVERT (this appendix)

QUIK.MOVE (this appendix)

TURN.AROUND (this appendix)

Global Variable

FLD.POINTER (1-D) INTEGER

This 1-dimensional array contains the temporary field pointers. This enables the user to access the obstacles by using the sequential order numbers given the fields on input.

TRAF.CONTROL (3-D) REAL (Appendix A)

Recursive Variable Integer

BRIDGE.STAT

Contains the status of the AVLB in the platoon.

COUNTER

Contains the number of five second move increments to get from the decision ellipse through the obstacle under the worst of conditions.

EQUIP.STAT

Contains the status of the blade tanks in the platoon.

I - Index for a do loop.

KNOW.LEVEL

Contains the value of the field attribute P6.FLD. This is the state of intelligence known about the obstacle.

0	no knowledge exists
2	knowledge exists
3	knowledge exists and a lane exists

OBS.FLD

The temporary field pointer of the obstacle.

OBS.NUM

The number of the obstacle to our front. This number is taken from the field's sequence on input.

Recursive Variable Real

DIST.AWAY

This is the distance to the obstacle from this decision ellipse.

INTERV

A constant term that is set so that element locations are updated every five seconds during a small portion of the simulation.

SMIN.OBS

Receives the value in meters of the semi-minor axis of the gap obstacle ellipse to the front.

Routines Called

JUNK.ALIVE (this appendix)

Temporary Attribute Integer

BYPASS

This variable is an argument for event DIVERT. It carries the route number that will cause lateral movement.

DIRC.ON.RT

Indicates whether or not a vehicle is moving forward or backward on his route.

0	Vehicle is moving in order of increasing MCP numbers along the route. (forward)
1	Vehicle is moving in order of decreasing MCP numbers along the route. (backward)

END.AREA

The number which identifies the ending movement area for this entity.

FLD.AKT

Code describing the kind of action for pending internal actions.

FLD.NO

Name of the field involved in any pending internal action.

FORMACODE

The formation number of the formation to be used by the platoon.

0 not in formation, vehicle moves along route without offset.

HOLD.FORM

This unit attribute is a storage place, where the units formation number for the platoon (FORMACODE) can be placed when lateral route movement is appropriate.

HOME

This variable is an argument for event DIVERT. It carries the MCP number that will cause lateral movement.

NEXT.MCP

Movement control point number (on the designated Route) toward which the element is now moving.

0 end of route has been reached

NUM.POINTER

This variable is an argument for event QUIK.MOVE. It carries the number of the entity to be moved.

REP.NUMBER

This variable is an argument for event QUIK.MOVE. It carries the number of iterations of scheduling left for this event to perform.

ROUTE

Indicates the number of the route along which the element is travelling.

0 not using a route

SIDE.STEPPER

This variable is an argument for event DIVERT. It carries the number of the entity to be moved laterally.

START.AREA

The number which identifies the starting movement area for this entity.

SYS.TYPE

This represents the general class of the system of the entity.

1 tanks
2 mounted infantry
3 dismounted infantry
4 artillery
5 air
6 air defense
7 bunker
8 coms/ew/acq/intel
9 other

TWIST

This variable is an argument for event TURN.AROUND. It carries the number of the entity to be turned in the opposite direction.

TYP.FLD

FIeld type code

1 - a mandatory dismount field
2 - a minefield
3 - a minefield decision field
4 - a tank ditch
5 - a road crater
6 - a blown bridge (short-wet gap)
7 - a gap decision field

Temporary Attribute Real

FLD.INT.DIST

This attribute of an entity contains the distance to a pending field internal action.

INT.TIME

This variable is an argument for event QUIK.MOVE. It carries the time interval number when the next event is to occur.

P1.FLD (appendix d)

P2.FLD (Appendix D)

P3.FLD (Appendix D)

P4.FLD (Appendix D)

P5.FLD (Appendix D)

P6.FLD (Appendix D)

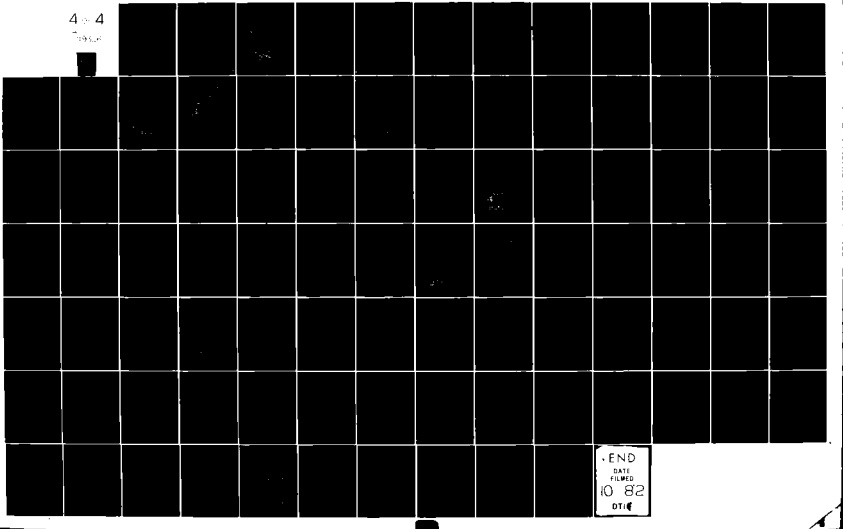
SAMIN.FLD

The semi-minor axis length of the elliptical field.

AD-A119 326 NAVAL POSTGRADUATE SCHOOL MONTEREY CA F/G 15/7
THE ENGINEER EFFECTS MODULE FOR THE STAR COMBAT MODEL.(U)
MAR 82 S C MAIN, J V MUDD

UNCLASSIFIED NL

4-4



END
DATE FILMED
10 82
DTIC

SPD

The entity's speed at the end of the most recent movement update.

XC.FLD

The X-coordinate of the center of an elliptical field.

YC.FLD

The Y-coordinate of the center of an elliptical field.

Brief Explanation

Lines 10-13 Interchanges the value of HOLD.FORM and FORMACODE. This allows the entity to pick upthe route formation if applicable.

Lines 15-19 Checks if the decision ellipse is on a bypass around the obstacle and takes appropriate actions if it is.

Lines 20-23 Dismounted infantry are not effected by gap type obstacles.

Lines 24-25 Links to the knowledge attribute, P6.FLD, of the obstacle to the front.

Lines 26-27 Turns the entity around if he is coming back from the obstacle.

Lines 28-31	Allows the user to stipulate if a bypass is a viable option or if the entity is going to "bull through".
Lines 34-40	Sets the counter variable to the appropriate number of QUIK.MOVE iterations needed to get to the gap obstacle.
Line 41	Checks on the status of the platoon's AVLBs and dozer blade tanks.
Lines 42-57	Allows the entity that has knowledge of a lane in the obstacle to move to the lane, if his platoon is void of breaching eqipment.
Lines 58-68	If the activating entity's platoon does not possess the required breaching equipment for the obstacle encountered and there is knowledge of the obstacle, then the entity can go to his bypass route and movement control point.
Line 70	Schedules five second movement updates for the entities going to the gap obstacle.

CODE

```
ROUTINE GAP.DECISION GIVEN A, FLD
DEFINE A,FLD,OBS.NUM,OBS.FLD,KNOW.LEVEL,EQUIP.STAT,
       I,BRIDGE.STAT, COUNTER AS INTEGER VARIABLES
DEFINE INTERV, SMIN.OBS, DIST.AWAY AS REAL VARIABLES
LET FLD.NO(A)=FLD
LET FLD.AKT(A)=TYP.FLD(FLD)
LET P4.FLD(FLD) = ROUTE(A)
LET P5.FLD(FLD) = NEXT.NCP(A)
LET INTERV = 5.0
IF HOLD.FORM(A) NE 999
   LET FORMACODE(A) = HOLD.FORM(A)
   LET HOLD.FORM(A) = 999
ALWAYS
''*****LINK TO OBSTACLE KNOWLEDGE ATTRIBUTE******
LET OBS.NUM = INT.F(P3.FLD(FLD))
IF OBS.NUM = 0
   ''THIS ELLIPSE CIRCUMVENTS THE OBSTACLE
   LET FLD.INT.DIST(A) = RINF.C
   RETURN
OTHERWISE
IF SYS.TYPE(A) = 3 ''DISMOUNTED INFANTRY
   LET FLD.INT.DIST(A) = RINF.C
   RETURN
OTHERWISE
LET OBS.FLD = FLD.POINTER(OBS.NUM)
LET KNOW.LEVEL = INT.F(P6.FLD(OBS.FLD))
IF DIRC.ON.RT(A) = 1
   SCHEDULE A TURN.AROUND GIVEN A NOW
ELSE
   IF P1.FLD(FLD) NE 0.0
      '' BYPASS EXISTS /FALSE-NO BYPASS(BULL-LANE)
      LET TRAF.CONTROL(OBS.NUM,ROUTE(A),2) =
                                          NEXT.NCP(A)
   ALWAYS
ALWAYS
LET FLD.INT.DIST(A) = RINF.C
IF P6.FLD(FLD) LE 0
    LET SMIN.OBS = SAMIN.FLD(OBS.FLD)
    LET DIST.AWAY =
            SQRT.F((XC.FLD(FLD)-XC.FLD(OBS.FLD))**2+
                   (YC.FLD(FLD)-YC.FLD(OBS.FLD))**2)
    LET COUNTER =
INT.F((INTERV+(DIST.AWAY+2*SMIN.OBS)/SPD(A))/INTERV)
    LET P6.FLD(FLD) = COUNTER
ALWAYS
CALL JUNK.ALIVE GIVEN A YIELDING
                                  EQUIP.STAT,BRIDGE.STAT
IF KNOW.LEVEL = 3
   IF TYP.FLD(OBS.FLD) EQ 4 ''TANK DITCH'' AND
      TRAF.CONTROL(OBS.NUM,ROUTE(A),3) NE RINF.C
      IF EQUIP.STAT NE 1 AND BRIDGE.STAT NE 1
         FOR I = 1 TO
            DIM.F(TRAF.CONTROL(OBS.NUM,*,*)), DO
            IF TRAF.CONTROL(OBS.NUM,I,3) EQ RINF.C
               AND TRAF.CONTROL(OBS.NUM,I,2) NE 0.0
               SCHEDULE A DIVERT(A,
                  INT.F(TRAF.CONTROL(OBS.NUM,I,1)),
                  INT.F(TRAF.CONTROL(OBS.NUM,I,2)))(
                                                  NOW
               RETURN
            OTHERWISE
         LOOP
         LET KNOW.LEVEL = 2
         ''LANE EXISTS-NOT PRIVY TO ITS LOCATION
      ALWAYS
```

```
56       ALWAYS
57    ALWAYS
58    IF KNOW.LEVEL = 2 ''WE KNOW THE OBSTACLE EXISTS
59       IF BRIDGE.STAT NE 1 ''NO BRIDGING EQUIPMENT
60          IF TYP.FLD(OBS.FLD) EQ 6 OR EQUIP.STAT NE 1
61             IF P1.FLD(FLD) NE 0.0''A BYPASS EXISTS
62                SCHEDULE A DIVERT(A,INT.F(P1.FLD(FLD)),
63                                  INT.F(P2.FLD(FLD))) NOW
64                RETURN
65             OTHERWISE
66          ALWAYS
67       ALWAYS
68    ALWAYS
69    LET COUNTER = INT.F(P6.FLD(FLD))
70    SCHEDULE A QUIK.MOVE GIVEN A,COUNTER,INTERV IN
                                                  INTERV UNITS
71    RETURN
72    END
```

19. Routine JUNK.ALIVE

Purpose

Routine JUNK.ALIVE checks the status of the platoon dozer blade tanks and vehicle launched bridges. This routine is called by routines GAP.DECISION, GAP.ENTRY, and events HEAVY.JUNK, and GAP.JOCK.

Given Arguments Integer

B

Pointer to the entity who's platoon status is being checked.

Yielding Arguments Integer

BRIDGE.STAT

Contains the status of the AVLBs in the platoon.

EQUIP.STAT

Contains the status of the blade tanks in the platoon.

Recursive Variable Integer

TNK

Pointer to an element in the platoon of the activating entity.

Set

PLT.UNIT (1-D)

This set is owned by the permanent entity PLATOON LEADER. It contains the list of temporary entities owned by this platoon.

Temporary Attribute Integer

BLADE.COND

This unit attribute gives the tank the ability to have a dozer blade characteristic.

0-no blade available
1-blade available but not in use
2-blade being used or vehicle is self breaching

KKILL

Indicates whether an entity has sustained a catastrophic kill.

0 no
1 yes

MKILL

Indicates whether an entity has sustained a mobility kill.

0 No
1 yes

PLT

The number of the platoon to which the entity belongs.

SYS.TYPE

This represents the general class of the system of the entity.

1 tanks
2 mounted infantry
3 dismounted infantry
4 artillery
5 air
6 air defense
7 bunker
8 comm/ew/acq/intel
9 other

WPN.TYPE

Describes the specific system within the system

code. For example, system type one is a tank and weapon type four for this system indicates an AVLB.

Brief Explanation

Lines 3-14 Checks the platoon for the presence of alive AVLBs and dozer blade tanks.

CODE

```
 1  ROUTINE JUNK.ALIVE GIVEN B
                          YIELDING EQUIP.STAT,BRIDGE.STAT
 2  DEFINE B,EQUIP.STAT,TNK,BRIDGE.STAT
          ! INTEGER VARIABLES
 3  FOR EVERY TNK IN PLT.UNIT(PLT(B)), DO
 4      IF KKILL(TNK) NE 1 AND HKILL(TNK) NE 1
 5          IF BLADE.COND(TNK) GE 1
 6              LET EQUIP.STAT = 1
 7          ALWAYS
 8          IF SYS.TYPE(TNK) EQ 1
 9              IF WPN.TYPE(TNK) = 4 OR WPN.TYPE(TNK) = 5
10                  LET BRIDGE.STAT = 1
11              ALWAYS
12          ALWAYS
13      ALWAYS
14  LOOP
15  RETURN
16  END
```

20. Routine GAP.ENTRY

Purpose

Routine GAP.ENTRY allows an entity to take the appropriate actions when encountering a gap type obstacle. This routine is called from Routine FLD.ACT.

Given Arguments Integer

A - Pointer to the entity that just entered the gap obstacle.

FLD- Pointer to the field in question.

Events Scheduled

GAP.JOCK (this appendix)

HALT (this appendix)

HEAVY.JUNK (this appendix)

TURN.AROUND (this appendix)

Global Variable Real

TRAP.CONTROL (3-D) (Appendix A)

Recursive Variable Integer

BRIDGE.STAT

Contains the status of the AVLBs in the platoon.

EQUIP.STAT

Contains the status of the blade tanks in the platoon.

KNOW.LEVEL

Contains the value of the field attribute P6.FLD. This is the state of intelligence known about the obstacle.

0	no knowledge exists
2	knowledge exists
3	knowledge exists and a lane exists

OBS.NUM

Receives the value of the field attribute NAM.FLD. This is the sequential id number in order of field creation.

TMK

Pointer to an element in the platoon of the activating entity.

Recursive Variable Real

SIT.REP.TIME

This is the time between status checks on the breaching operation.

SMIN

This variable is set to the semi-minor axis length of the elliptical field in meters.

Routines Called

HIDER

A routine used to determine the micro-terrain elevation for a selected element.

JUNK.ALIVE (this appendix)

Set

PLT.UNIT (1-D)

This set is owned by the permanent entity *PLATOON LEADER*. It contains the list of temporary entities owned by this platoon.

Temporary Attribute Integer

BLADE.COND

This unit attribute gives the tank the ability to have a dozer blade characteristic.

DEFNUM

The current position or activity of an element.

1	full defilade
2	turret defilade
3	firing defilade
4	half vehicle defilade
5	moving (defilade determined by the terrain model)
6	reached final area in movement

FLD.AKT

Code describing the kind of action for pending internal actions.

FLD.FORM

This unit attribute gives the entity the ability to change formations when encountering an obstacle.

FLD.NO

Name of the field involved in any pending internal action.

NAM.FLD

This is the sequential id number in order of field creation.

PLT

The number of the platoon to which the entity belongs.

ROUTE

Indicates the number of the route along which the element is travelling.

0 not using a route

SABOT.SHOOTER

This variable is an argument for event GAP.JOCK. It carries the number of the activating entity.

SLOW.DOWN

This variable is an argument for event HALT. It carries the number of the entity to be stopped.

STATUS

This variable is an argument for event HEAVY.JUNK. It carries the number of the entity to be updated.

SYS.TYPE

This represents the general class of the system of the entity.

1 tanks
2 mounted infantry
3 dismounted infantry
4 artillery
5 air
6 air defense
7 bunker
8 comm/ew/acq/intel
9 other

TWIST

This variable is an argument for event TURN.AROUND. It carries the number of the entity to be turned in the opposite direction.

TYP.FLD

Field type code.

1 - a mandatory dismount field
2 - a minefield
3 - a minefield decision field
4 - a tank ditch
5 - a road crater
6 - a blown bridge (short-wet gap)
7 - a gap decision field

WPN.TYPE

Describes the specific system within the system code. For example, system type 1 is a tank and weapon type four for this system indicates an AVLB.

Temporary Attribute Real

FLD.INT.DIST

This attribute of an entity contains the distance to a pending field internal action.

P1.FLD

This gap field attribute is a flag that specifies activation state.

0 not activated yet
1 this is an active gap obstacle

P6.FLD

This is the state of intelligence known about the obstacle.

0 no knowledge exists
2 knowledge exists
3 knowledge exists and a lane exists

P7.FLD

This minefield attribute contains the lane formation number.

SAMIN.FLD

This field attribute is the semi-minor axis length of the elliptical field in meters.

Brief Explanation

Lines 16-19 If the obstacle is not activated yet then the activating entity is allowed to move freely through this field.

Lines 20-23 Dismounted infantry soldiers are not affected by gap type obstacles.

Lines 24-27 When the entity is 200 meters from a large hole in the ground, he knows the gap exists.

Lines 28-36 If this route is on a cleared lane through the obstacle, change the formation of the platoon to a column and continue at present speed toward the crossing site.

Line 37 Checks the platoon AVLB and blade tank status.

Lines 41-43 If an entity has to "bull through" but a breaching operation is already

underway on his route, then the entity will stop and cover this operation by direct fire.

Lines 44-46 When an entity has to "bull through" and no breaching operations are ongoing for his route, then the entity will move to the gap and attempt a self breach.

Lines 47-56 If the entity has a bypass route but his platoon is void of appropriate breaching equipment forthis obstacle, then the activating entity and his platoon will turn around and move back toward the decision ellipse.

Lines 60-65 If the entity is an AVLB, then it will proceed to the gap to attempt a breach/span. The flag for an ongoing breach is set to the entity's platoon number.

Lines 66-70 If the entity is a blade tank and this obstacle is a road crater or tank ditch, then the tank will proceed to the gap to attempt a

breach. The flag for an ongoing breach is set to the blade tank's platoon number.

Lines 71-72 The vehicles in the platoon not equipped with breaching devices will stop and cover the gap clearing operation by direct fire from half defilade. A situation report on this operation is scheduled in the future.

CODE

```
 1  ROUTINE GAP.ENTRY GIVEN A,FLD
 2  DEFINE A,FLD,KNOW.LEVEL,OBS.NUM,TNK,EQUIP.STAT,
 3         BRIDGE.STAT AS INTEGER VARIABLES
 4  DEFINE SIT.REP.TIME,SMIN AS REAL VARIABLES
 5  SUBSTITUTE THESE 3 LINES FOR SUPPORT.SHOOT
 6    SCHEDULE A HALT GIVEN A NOW
 7    LET DEFNUM(A) = 4 ''HALF DEFILADE'' CALL HIDER(A)
 8    LET FLD.INT.DIST(A) = 50.0  ''A MUST DISTANCE
 9  LET FLD.NO(A) = FLD
10  LET FLD.AKT(A) = TYP.FLD(FLD)
11  LET KNOW.LEVEL = INT.F(P6.FLD(FLD))
12  LET OBS.NUM = NAM.FLD(FLD)
13  LET SIT.REP.TIME = 15.0  ''THE TIME OF STATUS CHECKS
14  LET TRAF.CONTROL(OBS.NUM,ROUTE(A),1) = ROUTE(A)
15  LET SMIN = SAMIN.FLD(FLD)
16  IF P1.FLD(FLD) EQ 0.0  '' NOT ACTIVATED YET
17     LET FLD.INT.DIST(A) = RINF.C
18     RETURN
19  OTHERWISE
20  IF SYS.TYPE(A) = 3 ''DISMOUNTED INFANTRY
21     LET FLD.INT.DIST(A) = RINF.C
22     RETURN
23  OTHERWISE
24  IF KNOW.LEVEL LE 1  ''NO KNOWLEDGE EXISTS
25     LET KNOW.LEVEL = 2
26     LET P6.FLD(FLD) = 2
27  ALWAYS  '' THEY KNOW THE OBSTACLE EXISTS
28     IF KNOW.LEVEL EQ 3
29        IF TYP.FLD(FLD) NE 4 ''NOT A TANK DITCH'' OR
30           (TYP.FLD(FLD) EQ 4 ''TANK DITCH'' AND
31           TRAF.CONTROL(OBS.NUM,ROUTE(A),3) EQ RINF.C)
32           LET FLD.INT.DIST(A) = SMIN - 50.0
33           LET FLD.FORM(A) = INT.F(P7.FLD(FLD))
34           RETURN
35        OTHERWISE
36     ALWAYS
37        CALL JUNK.ALIVE GIVEN A YIELDING
```

```
                                        EQUIP.STAT, BRIDGE.STAT
38          IF BRIDGE.STAT NE 1 ''NO BRIDGING EQUIPMENT
39             IF TYP.FLD(FLD) EQ 6 OR EQUIP.STAT NE 1
40               IF TRAF.CONTROL(OBS.NUM,ROUTE(A),2) = 0.0
41                IF TRAF.CONTROL(OBS.NUM,ROUTE(A),4) GT 0
42                   SUPPORT.SHOOT
43                   SCHEDULE A GAP.JOCK GIVEN A IN
                                        SIT.REP.TIME UNITS
44                ELSE
45                   LET FLD.INT.DIST(A) = SHIN - 50.0
46                ALWAYS
47               ELSE  ''A BYPASS EXISTS/B.B MUST HAVE ONE
48                   SCHEDULE A TURN.AROUND GIVEN A NOW
49                   FOR EVERY TNK IN PLT.UNIT(PLT(A))
50                            WITH SYS.TYPE(TNK) NE 3, DO
51                      IF FLD.NO(TNK) NE FLD
52                        SCHEDULE A TURN.AROUND(TNK) NOW
53                      ALWAYS
54                   LOOP
55                   LET FLD.INT.DIST(A) = RINF.C
56               ALWAYS
57               RETURN
58             OTHERWISE
59          ALWAYS
60          IF SYS.TYPE(A) EQ 1 AND
61                (WPN.TYPE(A) = 4 OR WPN.TYPE(A) = 5)
62            LET FLD.INT.DIST(A) = SHIN - 50.0
63            LET TRAF.CONTROL(OBS.NUM,ROUTE(A),4)= PLT(A)
64            RETURN
65          OTHERWISE
66          IF TYP.FLD(FLD) NE 6 AND BLADE.COND(A) EQ 1
67            LET FLD.INT.DIST(A) = SHIN - 50.0
68            LET TRAF.CONTROL(OBS.NUM,ROUTE(A),4)= PLT(A)
69            RETURN
70          OTHERWISE
71  SUPPORT.SHOOT
72  SCHEDULE A HEAVY.JUNK GIVEN A IN SIT.REP.TIME UNITS
73  RETURN
74  END
```

21. Routine GAP.INTERNAL

Purpose

Routine GAP.INTERNAL gives the activating entity the ability to take the appropriate actions at the location of a gap type obstacle. This routine is called from routine FLD.ACT.

Given Arguments Integer

A - Pointer to the entity that just activated this gap internal action.

Events Scheduled

DITCH.KILL	(this appendix)
GAP.BREACH	(this appendix)
HALT	(this appendix)
HEAVY.JUNK	(this appendix)
STAND.TO	(this appendix)
WALL.BREACH	(this appendix)

Global Variable Real

TRAP.CONTROL (3-D) (Appendix A)

Recursive Variable Integer

BRIDGE.STAT

Contains the status of the AVLBs in the platoon.

EQUIP.STAT

Contains the status of the blade tanks in the platoon.

FLD- Pointer to the field in question.

KNOW.LEVEL

Contains the value of the field attribute P6.FLD. This is the state of intelligence known about the obstacle.

0	no knowledge exists
2	knowledge exists
3	knowledge exists and a lane exists

OBS.NUM

Receives the value of the field attribute NAM.FLD. This is the sequential id number in order of field creation.

TNK

Pointer to an element in the platoon of the activating entity.

Recursive Variable Real

DRAW

Contains the value of a random number from a Uniform(0,1) distribution.

SIT.REP.TIME

This is the time between status checks on the breaching operation.

Routines Called

HIDER

A routine used to determine the micro-terrain elevation for a selected element.

Temporary Attribute Integer

BAGGED.BOY

This variable is an argument for event STAND.TO. It carries the number of the entity to be moved.

BLADE.COND

This unit attribute gives the tank the ability to have a dozer blade characteristic.

0-no blade available
1-blade available but not in use
2-blade being used or vehicle is self breaching

BREACHER

This variable is an argument for event GAP.BREACH. It carries the number of the entity that made the breach.

DEFNUM

The current position or activity of an element.

1 full defilade
2 turret defilade
3 firing defilade
4 half vehicle defilade
5 moving (defilade determined by the terrain model)
6 reached final area in movement

FLD.FORM

This unit attribute gives the entity the ability to change formations when encountering an obstacle.

FLD.NO

Name of the field involved in any pending internal action.

GULLY.CAT

This variable is an argument for event DITCH.KILL. It carries the number of the entity that is stuck in the gap.

HOLE

This variable is an argument for event GAP.BREACH. It carries the number of the field in question.

NAM.FLD

This is the sequential id number in order of field creation.

RAMMER

This variable is an argument for event WALL.BREACH. It carries the number of the entity that made the self breach.

ROUTE

Indicates the number of the route along which the element is travelling.

0 not using a route

SLOW.DOWN

This variable is an argument for event HALT. It carries the number of the entity to be stopped.

STATUS

This variable is an argument for event HEAVY.JUNK. It carries the number of the entity to be updated.

SYS.TYPE

This represents the general class of the system of the entity.

1 tanks
2 mounted infantry
3 dismounted infantry
4 artillery
5 air
6 air defense
7 bunker
8 comm/ew/acq/intel
9 other

TEAR.DOWN

This variable is an argument for event WALL.BREACH. It carries the number of the field in question.

TYP.FLD
Field type code.

1 - a mandatory dismount field
2 - a minefield
3 - a minefield decision field
4 - a tank ditch
5 - a road crater
6 - a blown bridge (short-wet gap)
7 - a gap decision field

WPN.TYPE

Describes the specific system within the system code. For example, system type 1 is a tank and weapon type four for this system indicates an AVLB.

Temporary Attribute Real

FLD.INT.DIST

This attribute of an entity contains the distance to a pending field internal action.

FSPEED.FAC

This unit attribute gives the entity the ability to slow down because of obstacle encounter.

P6.FLD

Knowledge level of this obstacle.

0	no knowledge exists
2	knowledge exists
3	knowledge exists and a lane exists

P7.FLD

This minefield attribute contains the lane formation number.

P8.FLD

Lane speed factor.

P9.FLD

AVLB activation delay time in seconds.

P10.FLD

Tank dozer breach time in seconds.

P11.FLD

Tank self breach time in seconds.

P12.FLD

Probability of a self breach getting stuck in ditch.

Brief Explanation

Lines 10-18 The entity's speed is reduced and he is put into a column formation. This simulates the funneling effect of the actual crossing.

Lines 19-42 Simulates the self-breach option against road craters and tank ditches.

Lines 23-26 When an entity has to self breach a tank ditch, he checks if a breach has been accomplished elsewhere. If a lane exists then the entity's speed is reduced substantially to simulate his movement to another existing route/lane. His formation is also changed to column.

Lines 28-40 Simulates the actual self breach. The entity is stopped, he is put in gun tube defilade (in ditch), and he is prevented from firing in this

configuration. An input parameter is Monte Carlo'ed against, to determine if the entity is going to be a mobility casualty. When not stuck in the ditch the entity has a breach scheduled to occur in the future.

Line 43 — Schedules a breaching operation status update for the AVLB's and blade tank's attempting to breach this obstacle.

Lines 44-48 — Stops the blade tank and starts his breaching effort. This tank is put in half defilade (blade and ditch shading), and he is prevented from firing. A successful breach is scheduled to occur in the future.

Lines 49-51 — Stops the AVLB and starts his breaching effort. A successful breach/span is scheduled to occur in the future.

CODE

```
1  ROUTINE GAP.INTERNAL GIVEN A
2  DEFINE A,FLD,KNOW.LEVEL,OBS.NUM,EQUIP.STAT,
3         BRIDGE.STAT,TNK AS INTEGER VARIABLES
4  DEFINE DRAW,SIT.REP.TIME AS REAL VARIABLES
```

```
LET FLD = FLD.NO(A)
LET KNOW.LEVEL = INT.F(P6.FLD(FLD))
LET OBS.NUM = NAM.FLD(FLD)
LET SIT.REP.TIME = 15.0   ''TIME TO UPDATE STATUS
LET FLD.INT.DIST(A) = 50.0  ''A MUST DISTANCE
IF KNOW.LEVEL EQ 3
   IF TYP.FLD(FLD) NE 4 OR (TYP.FLD(FLD) EQ 4 AND
         TRAF.CONTROL(OBS.NUM,ROUTE(A),3) EQ 2INF.C)
      LET FSPEED.FAC(A) = P8.FLD(FLD)
      LET FLD.INT.DIST(A) = RINF.C
      LET FLD.FORM(A) = INT.F(P7.FLD(FLD))
      RETURN
   OTHERWISE
ALWAYS  ''WE NEED TO BREACH A LANE THRU
IF TYP.FLD(FLD) NE 6
   IF BLADE.COND(A) NE 1
      IF SYS.TYPE(A) NE 1 OR (SYS.TYPE(A) EQ 1 AND
           (WPN.TYPE(A) NE 4 AND WPN.TYPE(A) NE 5))
         IF TYP.FLD(FLD) EQ 4 AND KNOW.LEVEL EQ 3
            LET FSPEED.FAC(A) = P8.FLD(FLD)*.5
            LET FLD.INT.DIST(A) = RINF.C
            LET FLD.FORM(A) = INT.F(P7.FLD(FLD))
         ELSE
            SCHEDULE A HALT GIVEN A NOW
            LET DEFNUM(A) = 3 ''GUN'' CALL HIDER(A)
            LET BLADE.COND(A) = 2   ''STOP SHOOTING
            LET DRAW = UNIFORM.F(0.,1.,7)
            IF P12.FLD(FLD) GT DRAW
               SCHEDULE A DITCH.KILL(A) IN
                                 (P11.FLD(FLD)/2.0) UNITS
            ELSE
               SCHEDULE A WALL.BREACH(A,FLD) IN
                                       P11.FLD(FLD) UNITS
               SCHEDULE A STAND.TO(A) IN
                                    (P11.FLD(FLD)+1) UNITS
            ALWAYS
         ALWAYS
         RETURN
      OTHERWISE
   ALWAYS
ALWAYS
SCHEDULE A HEAVY.JUNK GIVEN A IN SIT.REP.TIME UNITS
IF BLADE.COND(A) EQ 1
  SCHEDULE A HALT GIVEN A NOW
  LET DEFNUM(A) = 4 ''HALF DEFILADE'' CALL HIDER(A)
  LET BLADE.COND(A) = 2   ''STOP SHOOTING
  SCHEDULE A GAP.BREACH(A,FLD) IN P10.FLD(FLD) UNITS
ELSE
  SCHEDULE A HALT GIVEN A NOW
  SCHEDULE A GAP.BREACH(A,FLD) IN P9.FLD(FLD) UNITS
ALWAYS
RETURN
END
```

22. Routine GAP.LEAVE

Purpose

Routine GAP.LEAVE performs the gap obstacle exit actions for the activating/moving entity. This routine is called from routine FLD.ACT.

Given Arguments Integer

A - Pointer to the entity that just activated this gap exit action.

Global Variable Real

TRAF.CONTROL (3-D) (Appendix A)

Recursive Variable Integer

FLD- Pointer to the field in question.

OBS.NUM

Receives the value of the field attribute NAM.FLD. This is the sequential id number in order of field creation.

Temporary Attribute Integer

BLADE.COND

This unit attribute gives the tank the ability to have a dozer blade characteristic.

0-no blade available
1-blade available but not in use
2-blade being used or vehicle is self breaching

DIRC.ON.RT

Indicates whether or not a vehicle is moving forward or backward on his route.

0 Vehicle is moving in order of increasing MCP numbers along the route. (forward)

1 Vehicle is moving in order of decreasing MCP numbers along the route. (backward)

FLD.FORM

This unit attribute gives the entity the ability to change formations when encountering an obstacle.

FLD.NO

Name of the field involved in any pending internal action.

NAM.FLD

This is the sequential id number in order of field creation.

ROUTE

Indicates the number of the route along which the element is travelling.

0 not using a route

SYS.TYPE

This represents the general class of the system of the entity.

1 tanks
2 mounted infantry
3 dismounted infantry
4 artillery
5 air
6 air defense
7 bunker
8 comm/ew/acq/intel
9 other

Temporary Attribute Real

PSPEED.FAC

This unit attribute gives the entity the ability to slow down because of obstacle encounter.

P1.FLD

This gap field attribute is a flag that specifies activation state.

0	not activated yet
1	this is an active gap obstacle

P6.FLD

Knowledge level of this obstacle.

0	no knowledge exists
2	knowledge exists
3	knowledge exists and a lane exists

Brief Explanation

Lines 3-5	Dismounted infantry are not affected.
Lines 7-9	Inactivated obstacles have no effects.
Lines 10-12	Prevents unplanned entities from affecting the situation.
Lines 13-15	Reinitializes gap obstacle related vehicle attributes.
Lines 16-18	Changes the knowledge level of this obstacle. This transmits to follow-on units that a breach/span exists in this obstacle.

CODE

```
 1  ROUTINE GAP.LEAVE GIVEN A
 2  DEFINE A,FLD,OBS.NUM AS INTEGER VARIABLES
 3  IF SYS.TYPE(A) = 3 ''DISMOUNTED INFANTRY
 4     RETURN
 5  OTHERWISE
 6  LET FLD = FLD.NO(A)
 7  IF P1.FLD(FLD) EQ 0.0  ''NOT ACTIVATED YET
 8     RETURN
 9  OTHERWISE
10  IF BLADE.COND(A) GE 2
11     RETURN
12  OTHERWISE
13  LET OBS.NUM = NAM.FLD(FLD)
14  LET FSPEED.FAC(A) = 1.0
15  LET FLD.FORM(A) = 0
16  IF DIRC.ON.RT(A) = 0
17     LET P6.FLD(FLD) = 3
18  ALWAYS
19  RETURN
20  END
```

23. Event DITCH.KILL

Purpose

Event DITCH.KILL inflicts a mobility kill on the entity that got stuck in the tank ditch or road crater while attempting a self breach. This event is scheduled from routine GAP.INTERNAL.

Given Arguments Integer

B - The pointer to the entity that is stuck in the gap obstacle.

Global Variable Integer

DAM.ARRAY (1-D)

Contains the possible values for HIT.STATE.

1	NDAM
2	MDAM
3	HDAM
4	MFDM
5	DEAD
6	MISS

DAMAGE.NUM

Indicates the damage status of an entity after having a round impact on or near it.

1	hit but already MF killed
2	mobility damage
3	firepower damage
4	mobility and firepower damage
5	catastrophic kill
6	miss

DEAD.ATKR

Tallies the number of red casualties.

ONDISK

Indicates whether or not the user desires the shot list and final attribute list to user specified disk files.

0	no, paper output only
1	yes

YES

Indicates a value of 1. Used in conditional statements.

Recursive Variable Integer

WHOCALLED

A flag used by several routines to determine appropriate actions and executions.

Recursive Variable Real

EFKILL

The probability of at least a firepower kill.

EMKILL

The probability of at least a mobility kill.

EMNFKILL

The union of EMKILL and EFKILL.

KAYKILL

The probability of a catastrophic kill.

Routines Called

ATRIT

A Routine which assesses the attrition against a target which has been hit. It checks for a catastrophic kill, increments mobility and firepower damage, and checks the FKILL, MKILL, and MFKILL levels.

HIDER

A routine used to determine the micro-terrain elevation for a selected element.

Temporary Attribute Alpha

HIT.STATE

An alpha variable indicating whether or not an element is alive or dead.

Temporary Attribute Integer

COLOR

Indicates the color of the element.

0	red (attacker)
1	blue (defender)

DEFNUM

The current position or activity of an element.

1	full defilade
2	turret defilade
3	firing defilade
4	half vehicle defilade
5	moving (defilade determined by the terrain model)
6	reached final area in movement

FIRED.AT

Indicates the total number of rounds fired at an entity.

FKILL

Indicates whether an entity has sustained a firepower kill.

0 no
1 yes

GULLY.CAT

This variable is an argument for event DITCH.KILL. It carries the number of the entity that is stuck in the gap.

K.HIT

Indicates the number of hits sustained by an entity which were sufficient to cause a catastrophic kill.

KKILL

Indicates whether an entity has sustained a catastrophic kill.

0 no
1 yes

MFKILL

Indicates whether an entity has sustained a simultaneous mobility and firepower kill.

0 no
1 yes

MKILL

Indicates whether an entity has sustained a mobility kill.

0 no
1 yes

NAME

The element number of the entity.

NUM.HIT

Indicates the total number of hits sustained by an entity. This includes no damage hits.

ROUTE

Indicates the number of the route along which the element is travelling.

0 not using a route

WPN.TYPE

Describes the specific system within the system code. For example, system type 1 is a tank and weapon type four for this system indicates an AVLB.

Temporary Attribute Real

F.D

Indicates the accumulated percentage of firepower damage sustained by the entity.

M.D

Indicates the accumulated percentage of mobility damage sustained by the entity.

SPD

The entity's speed at the end of the most recent movement update.

X.CURRENT

The X-coordinate for the entity as of the last movement update.

Y.CURRENT

The Y-coordinate for the entity as of the last movement update.

Z.CURRENT

The elevation for the entity as of the last movement update.

Brief Explanation

Lines 5-7 The entity is already dead, do nothing.

Lines 8-33 The entity is a mobility casualty. This happened half-way through his self breach/bull through effort in the ditch. The killer/victim scoreboard is updated and the entity is put in turret defilade (bottom of ditch); still unable to fire. .

CODE

```
 1  EVENT DITCH.KILL GIVEN B
 2  DEFINE B,WHOCALLED AS INTEGER VARIABLES
 3  DEFINE EHKILL,EPKILL,EHNPKILL,KAYKILL
          AS REAL VARIABLES
 4  LET WHOCALLED = 1
 5  IF KKILL(B) EQ 1
 6     RETURN
 7  OTHERWISE
 8  LET EHKILL = 1.0
 9  LET EPKILL = 0.0
10  LET EHNPKILL = 1.0
11  LET KAYKILL = 0.0
12  CALL ATRIT(B,B,EHKILL,EPKILL,EHNPKILL,KAYKILL,
                                        WHOCALLED)
13  LET HIT.STATE(B) = DAM.ARRAY(DAMAGE.NUM)
14  USE UNIT 6 FOR OUTPUT
15     START NEW LINE
16  'UP'
17  WRITE  NAME(B), WPN.TYPE(B), TIME.V, X.CURRENT(B),
18  Y.CURRENT(B), Z.CURRENT(B), SPD(B), DEFNUM(B),
    HIT.STATE(B), H.D(B)
19  F.D(B), EKILL(B), PKILL(B), NPKILL(B), KKILL(B),
20  K.HIT(B), FIRED.AT(B), NUM.HIT(B)
21  AS  S 6, 4GAP 4, S 12, 2 I 4, I 5, S 5, S 22, 3 I 7,
       D(5,1), I 3,
22     S 5, S 6, S 1, A 4, 2 D(5,2), 2 I 2, 5 I 3
```

```
23   IF ONDISK=YES AND WRITE.V=6 USE UNIT 10 FOR OUTPUT
                                 START NEW LINE GO TO UP ELSE
24     IF COLOR(B) EQ 0  AND ( DAMAGE.NUM EQ 5 OR
                                  ((DAMAGE.NUM EQ 4 OR
25       DAMAGE.NUM EQ 3 OR DAMAGE.NUM EQ 2 ) AND
26     (   MFKILL(B) EQ 1)))
27    AND ( DAMAGE.NUM NE 5 OR MFKILL(B) NE 1 )
28      AND HIT.STATE (B) NE "DEAD" AND
                                       HIT.STATE(B) NE "ADED"
29   ADD 1 TO DEAD.ATKR ALWAYS
30   IF HIT.STATE(B) NE "DEAD" AND HIT.STATE(B) NE "ADED"
31   LET HIT.STATE(B)="    "    ALWAYS
32   START NEW LINE
33   LET DEFNUM(B) = 2 ''TURRET DEFILADE'' CALL HIDER(B)
34   RETURN
35   END
```

24. Event WALL.BREACH

Purpose

Event WALL.BREACH simulates the successful self breach of a road crater or tank ditch. This event is scheduled from routine GAP.INTERNAL.

Given Arguments Integer

B - The pointer to the entity that has self breached the gap obstacle.

FLD- Pointer to the field in question.

Events Scheduled

QUIK.MOVE (this appendix)

Global Variable Real

LIM.SPDS (3-D)

Indicates the limiting speeds for vehicles in a mounted attack.

TRAF.CONTROL (3-D) (Appendix A)

Recursive Variable Integer

COUNTER

Contains the number of five second move increments that will take place for this entity. This movement update is accomplished by scheduling event QUIK.MOVE.

OBS.NUM

Receives the value of the field attribute NAM.FLD. This is the sequential id number in order of field creation.

Recursive Variable Real

INTERV

A constant term that is set so that element locations are updated every five seconds during a small portion of the simulation.

Routines Called

HIDER

A routine used to determine the micro-terrain elevation for a selected element.

Temporary Attribute Integer

BLADE.COND

This unit attribute gives the tank the ability to have a dozer blade characteristic.

0-no blade available
1-blade available but not in use
2-blade being used or vehicle is self breaching

DEFNUM

The current position or activity of an element.

1 full defilade
2 turret defilade
3 firing defilade
4 half vehicle defilade
5 moving (defilade determined by the terrain model)
6 reached final area in movement

FLD.NO

Name of the field involved in any pending internal action.

KKILL

Indicates whether an entity has sustained a catastrophic kill.

0 no
1 yes

MKILL

Indicates whether an entity has sustained a mobility kill.

0 no
1 yes

NAM.FLD

This is the sequential id number in order of field creation.

NUM.POINTER

This variable is an argument for event QUIK.MOVE. It carries the number of the entity to be moved.

RAMMER

This variable is an argument for event WALL.BREACH. It carries the number of the entity that made the self breach.

REP.NUMBER

This variable is an argument for event QUIK.MOVE. It carries the number of iterations of scheduling left for this event to perform.

ROUTE

Indicates the number of the route along which the element is travelling.

0 not using a route

SYS.TYPE

This represents the general class of the system of the entity.

1 tanks

2	mounted infantry
3	dismounted infantry
4	artillery
5	air
6	air defense
7	bunker
8	comm/ew/acq/intel
9	other

TEAR.DOWN

This variable is an argument for event WALL.BREACH. It carries the number of the field in question.

WPN.TYPE

Describes the specific system within the system code.

Temporary Attribute Real

FLD.INT.DIST

This attribute of an entity contains the distance to a pending field internal action.

P6.FLD

Knowledge level of this obstacle.

0	no knowledge exists
2	knowledge exists
3	knowledge exists and a lane exists

INT.TIME

This variable is an argument for event QUIK.MOVE. It carries the time interval number when the next event is to be performed.

SAMIN.FLD

This field attribute is the semi-minor axis length of the elliptical field in meters.

Brief Explanation

Lines 6-8 Prevents unplanned entities from affecting the the situation.

Lines 10-12 Determines the appropriate number of five second movement updates to perform to exit this obstacle.

Lines 13-17 Changes the knowledge level to three, a breach exists on this route. It takes the self breacher out of the ditch and lets him shoot again.

CODE

```
 1  EVENT WALL.BREACH GIVEN B,FLD
 2  DEFINE B,FLD,OBS.NUM,COUNTER AS INTEGER VARIABLES
 3  DEFINE INTERV AS A REAL VARIABLE
 4  LET OBS.NUM = NAM.FLD(FLD)
 5  LET INTERV = 5.0   ''MOVE UPDATES
 6  IF FLD.NC(B) NE FLD
 7     RETURN
 8  OTHERWISE
 9  IF KKILL(B) NE 1 AND MKILL(B) NE 1
10     LET COUNTER = INT.F(SAMIN.FLD(FLD)/
11        (.5*LIN.SPDS(SYS.TYPE(B),WPN.TYPE(B),4))/INTERV)
12     SCHEDULE A QUIK.MOVE(B,COUNTER,INTERV) NOW
13     LET P6.FLD(FLD) = 3.0
14     LET DEFNUM(B) = 5 ''MOVING'' CALL HIDER(B)
15     LET BLADE.COND(B) = 0
16     LET TRAF.CONTROL(OBS.NUM,ROUTE(B),3) = RINF.C
17     LET FLD.INT.DIST(B) = RINF.C
18  ALWAYS
19  RETURN
20  END
```

25. Event GAP.BREACH

Purpose

Event GAP.BREACH simulates the successful breaching of a gap type obstacle by either a vehicle launched bridge or a tank mounted dozer blade. This event is scheduled from routine GAP.INTERNAL.

Given Arguments Integer

B - The Pointer to the entity that has breached the gap obstacle.

FLD- Pointer to the field in question.

Events Scheduled

QUIK.MOVE (this appendix)

STAND.TO (this appendix)

Global Variable Real

LIM.SPDS (3-D)

Indicates the limiting speeds for vehicles in a mounted attack.

TRAF.CONTROL (3-D) (Appendix A)

Recursive Variable Integer

COUNTER

Contains the number of five second move increments that will take place for this entity. This movement update is accomplished by scheduling event QUIK.MOVE.

OBS.NUM

Receives the value of the field attribute NAM.FLD. This is the sequential id number in order of field creation.

TNK

Pointer to an element in the platoon of the activating entity.

Recursive Variable Real

INTERV

A constant term that is set so that element locations are updated every five seconds during a small portion of the simulation.

Routines Called

HIDER

A routine used to determine the micro-terrain elevation for a selected element.

Set

PLT.UNIT (1-D)

This set is owned by the permanent entity PLATOON LEADER. It contains the list of temporary entities owned by this platoon.

Temporary Attribute Integer

BAGGED.BOY

This variable is an argument for event STAND.TO. It carries the number of the entity to be moved.

BLADE.COND

This unit attribute gives the tank the ability to have a dozer blade characteristic.

0-no blade available
1-blade available but not in use
2-blade being used or vehicle is self breaching

BREACHER

This variable is an argument for event GAP.BREACH. It carries the number of the entity that made the breach.

DEFNUM

The current position or activity of an element.

1 full defilade
2 turret defilade
3 firing defilade
4 half vehicle defilade
5 moving (defilade determined by the terrain model)
6 reached final area in movement

FLD.FORM

This unit attribute gives the entity the ability to change formations when encountering an obstacle.

FLD.NO

Name of the field involved in any pending internal action.

HOLE

This variable is an argument for event GAP.BREACH. It carries the number of the field in question.

KKILL

Indicates whether an entity has sustained a catastrophic kill.

0 no
1 yes

MKILL

Indicates whether an entity has sustained a mobility kill.

0 no
1 yes

MV.STATE

The primary control variable for initiating and stopping movement.

0 In position, do not move.
1 First call to move, do a route select and start to move.
2 Continue movement along a previously selected route.
3 Stop along the route.
4 Next position has been reached, so stop.
5 Final position has been reached, never move again.

NAM.FLD

This is the sequential id number in order of field creation.

NUM.POINTER

This variable is an argument for event QUIK.MOVE. It carries the number of the entity to be moved.

PLT

The number of the platoon to which the entity belongs.

REP.NUMBER

This variable is an argument for event QUIK.MOVE. It carries the number of iterations or scheduling left for this event to perform.

ROUTE

Indicates the number of the route along which the element is travelling.

0 not using a route

SYS.TYPE

This represents the general class of the system of the entity.

1 tanks
2 mounted infantry
3 dismounted infantry
4 artillery
5 air
6 air defense
7 bunker
8 comm/ew/acq/intel
9 other

WPN.TYPE

Describes the specific system within the system code. For example, system type 1 is a tank and weapon type four for this system indicates an AVLB.

Temporary Attribute Real

FLD.INT.DIST

This attribute of an entity contains the distance to a pending field internal action.

FSPEED.FAC

This unit attribute gives the entity the ability to slow down because of obstacle encounter.

INT.TIME

This variable is an argument for event QUIK.MOVE. It carries the time interval number when the next event is to be performed.

P6.FLD

Knowledge level of this obstacle.

0	no knowledge exists
2	knowledge exists
3	knowledge of a lane exists

P7.FLD

This minefield attribute contains the lane formation number.

P8.FLD

Lane speed factor.

SAMIN.FLD

This field attribute is the semi-minor axis length of the elliptical field in meters.

Brief Explanation

Lines 6-8 Does nothing if the entity has started to move again or if he has left this obstacle. These actions could have been produced as a result of event HEAVY.JUNK actions.

Lines 10-13 Takes the bridge launcher out of the simulation when its bridge is laid over the gap.

Lines 15-17 If the blade tank that scheduled this event has moved because of event

HEAVY.JUNK actions, then this event does not effect his present status.

Lines 19-33 Starts the platoon moving again and puts them in a column formation in order to cross through/ over the gap obstacle. The blade tanks that were breaching are slowed down to allow the rest of their platoon to catch up. Enough five second move increments are scheduled to upgrade the detail of the crossing and exit of this field.

Lines 34-35 The knowledge level of this obstacle is changed so that follow-on units can know that a breach exists on this route.

Line 36 The flag indicating an ongoing breaching operation is turned off.

CODE

```
 1  EVENT GAP.BREACH GIVEN B,FLD
 2  DEFINE B,FLD,TNK,OBS.NUM,COUNTER
          AS INTEGER VARIABLES
 3  DEFINE INTERV AS A REAL VARIABLE
 4  LET OBS.NUM = MAN.FLD(FLD)
 5  LET INTERV = 5.0   ''MOVE UPDATES
 6  IF MV.STATE(B) NE 3 OR FLD.NO(B) NE FLD
 7     RETURN
 8  OTHERWISE
 9  IF MKILL(B) NE 1 AND MKILL(B) NE 1
```

```
10      IF SYS.TYPE(B) EQ 1 AND
                  (WPN.TYPE(B) EQ 4 OR WPN.TYPE(B) EQ 5)
11         LET MV.STATE(B) = 5   ''LAUNCHED BRIDGE OUT
12         LET DEFNUM(B) = 1 ''FULL'' CALL HIDER(B)
13         SCHEDULE A QUIK.MOVE(B,0,0.0) NOW   ''TO MOVE
14      ELSE
15         IF BLADE.COND(B) NE 2
16            RETURN
17         OTHERWISE
18      ALWAYS
19      FOR EVERY TNK IN PLT.UNIT(PLT(B)) WITH
                                    SYS.TYPE(TNK) NE 3, DO
20        IF KKILL(TNK) NE 1 AND MKILL(TNK) NE 1 AND
                                    MV.STATE(TNK) NE 5
21           SCHEDULE A STAND.TO GIVEN TNK NOW
22           LET DEFNUM(TNK) = 5 ''GO'' CALL HIDER(TNK)
23           LET FLD.INT.DIST(TNK) = SAMIN.FLD(FLD)-50.0
24           LET FLD.FORM(TNK) = INT.F(P7.FLD(FLD))
25           IF BLADE.COND(TNK) GT 0
26              LET FSPEED.FAC(TNK) = (P8.FLD(FLD)+1)/4
27              LET BLADE.COND(TNK) = 1
28           ALWAYS
29           LET COUNTER = INT.F((SAMIN.FLD(FLD)*2.0)/
30            (.5*LIM.SPDS(SYS.TYPE(TNK),WPN.TYPE(TNK),4))
                                                /INTERV)
31           SCHEDULE A QUIK.MOVE(TNK,COUNTER,INTERV) NOW
32        ALWAYS
33      LOOP
34      LET P6.FLD(FLD) = 3.0
35      LET TRAF.CONTROL(OBS.NUM,ROUTE(B),3) = RINF.C
36      LET TRAF.CONTROL(OBS.NUM,ROUTE(B),4) = 0.0
37   ALWAYS
38   RETURN
39   END
```

26. Event HEAVY.JUNK

Purpose

Event HEAVY.JUNK gives a situation report on the gap breaching operation in progress. This event also initiates the appropriate actions depending on the status update. Event HEAVY.JUNK is scheduled from routines GAP.ENTRY, GAP.INTERNAL, and event HEAVY.JUNK.

Given Arguments Integer

B - The pointer to the entity that has activated this status check of breaching equipment.

Events Scheduled

QUIK.MOVE (this appendix)

STAND.TO (this appendix)

TURN.AROUND (this appendix)

Global Variable Real

TRAF.CONTROL (3-D) (Appendix A)

LIM.SPDS (3-D)

Indicates the limiting speeds for vehicles in a mounted attack.

Recursive Variable Integer

BRIDGE.STAT

Contains the status of the AVLBs in the platoon.

COUNTER

Contains the number of five second move increments that will take place for this entity. This movement update is accomplished by scheduling event QUIK.MOVE.

EQUIP.STAT

Contains the status of the blade tanks in the platoon.

FLD- Pointer to the field in question.

KNOW.LEVEL

Contains the value of the field attribute P6.FLD. This is the state of intelligence known about the obstacle.

0	no knowledge exists
2	knowledge exists
3	knowledge exists and a lane exists

OBS.NUM

Receives the value of the field attribute NAM.FLD. This is the sequential id number in order of field creation.

TNK

Pointer to an element in the platoon of the activating entity.

Recursive Variable Real

INTERV

A constant term that is set so that element

locations are updated every five seconds during a small portion of the simulation.

SIT.REP.TIME

This is the time between status checks on the breaching operation.

Routines Called

HIDER

A routine used to determine the micro-terrain elevation for a selected element.

JUNK.ALIVE (this appendix)

Set

PLT.UNIT (1-D)

This set is owned by the permanent entity PLATOON LEADER. It contains the list of temporary entities owned by this platoon.

Temporary Attribute Integer

BAGGED.BOY

This variable is an argument for event STAND.TO. It carries the number of the entity to be moved.

BLADE.COND

This unit attribute gives the tank the ability to have a dozer blade characteristic.

0-no blade available
1-blade available but not in use
2-blade being used or vehicle is self breaching

DEFNUM

The current position or activity of an element.

1	full defilade
2	turret defilade
3	firing defilade
4	half vehicle defilade
5	moving (defilade determined by the terrain model)
6	reached final area in movement

FLD.FORM

This unit attribute gives the entity the ability to change formations When encountering an obstacle.

FLD.NO

Name of the field involved in any pending internal action.

KKILL

Indicates whether an entity has sustained a catastrophic kill.

0	no
1	yes

MKILL

Indicates whether an entity has sustained a mobility kill.

0	no
1	yes

MV.STATE

The primary control variable for initiating and stopping movement.

0	In position, do not move.
1	First call to move, do a route select and start to move.
2	Continue movement along a previously selected route.
3	Stop along the route.
4	Next position has been reached, so stop.
5	Final position has been reached, never move again.

NAM.FLD

This is the sequential id number in order of field creation.

NUM.POINTER

This variable is an argument for event QUIK.MOVE. It carries the number of the entity to be moved.

PLT- The number of the platoon to which the entity belongs.

REP.NUMBER

This variable is an argument for event QUIK.MOVE. It carries the number of iterations of scheduling left for this event to perform.

ROUTE

Indicates the number of the route along which the element is travelling.

0 not using a route

STATUS

This variable is an argument for event HEAVY.JUNK. It carries the number of the entity to be updated.

SYS.TYPE

This represents the general class of the system of the entity.

1 tanks
2 mounted infantry
3 dismounted infantry
4 artillery
5 air
6 air defense
7 bunker
8 comm/ew/acq/intel
9 other

TWIST

This variable is an argument for event TURN.AROUND. It carries the number of the entity to be turned in the opposite direction.

TYP.FLD

Field type code.

1 - a mandatory dismount field
2 - a minefield
3 - a minefield decision field
4 - a tank ditch
5 - a road crater
6 - a blown bridge (short-wet gap)
7 - a gap decision field

WPN.TYPE

Describes the specific system within the system code.

Temporary Attribute Real

FLD.INT.DIST

This attribute of an entity contains the distance to a pending field internal action.

FSPEED.FAC

This unit attribute gives the entity the ability to slow down because of obstacle encounter.

INT.TIME

This variable is an argument for event QUIK.MOVE. It carries the time interval number when the next event is to occur.

P6.FLD

Knowledge level of this obstacle.

0 no knowledge exists
2 knowledge exists

3 knowledge of a lane exists

P7.FLD

This minefield attribute contains the lane formation number.

P8.FLD

Lane speed factor.

SAMIN.FLD

This field attribute is the semi-minor axis length of the elliptical field in meters.

T.SPD

The simulation time at which the most recent movement update ended. (the time that SPD was last set.)

Brief Explanation

Lines 11-34 When a breach has been successfully performed, all elements on this cleared route continue to move or start moving again, depending on their current move status. These elements are put into column formations and the breaching equipment is slowed down to allow the rest of the platoon time to catch up.

Line 35 Checks on the current status of the

	platoon vehicles performing the breach.
Lines 36-62	Simulates the actions that happen when the breaching equipment at the gap is depleted from direct/indirect fire.
Lines 38-45	If a bypass does not exist, the platoon's remaining vehicles start moving to the gap in order to attempt a self breach. The flag for an ongoing breach is turned off.
Lines 46-59	When the bypass option is still a viable one and the breaching operation has failed, then the platoon is turned around and started back toward the decision ellipse.
Line 63	If the operation is still being performed by alive platoon equipment, then another breaching operation situation report is scheduled for the future.

CODE

```
 1  EVENT HEAVY.JUNK GIVEN B
 2  DEFINE B,FLD,TNK,KNOW.LEVEL,OBS.NUM,EQUIP.STAT,
 3         BRIDGE.STAT,COUNTER AS INTEGER VARIABLES
 4  DEFINE SIT.REP.TIME,INTERV AS REAL VARIABLES
 5  LET FLD = FLD.NO(B)
 6  LET KNOW.LEVEL = INT.F(P6.FLD(FLD))
 7  LET INTERV = 5.0   ''MOVE UPDATES
 8  LET SIT.REP.TIME = 15.0   ''TIME TO UPDATE STATUS
 9  LET OBS.NUM = NAM.FLD(FLD)
10  IF KKILL(B) NE 1 AND MKILL(B) NE 1 AND
                                        MV.STATE(B) NE 5
11      IF KNOW.LEVEL EQ 3
12          IF TYP.FLD(FLD) NE 4 OR (TYP.FLD(FLD) EQ 4 AND
13             TRAF.CONTROL(OBS.NUM,ROUTE(B),3) EQ RINF.C)
14             IF MV.STATE(B) NE 2
15              LET T.SPD(B) = TIME.V
16              LET COUNTER = INT.F((SAMIN.FLD(FLD)*2.0)/
17                (.5*LIN.SPDS(SYS.TYPE(B),WPN.TYPE(B),4))
                                                 /INTERV)
18              SCHEDULE A QUIK.MOVE(B,COUNTER,INTERV) NOW
19             ALWAYS
20             LET MV.STATE(B) = 2   ''MOVING
21             LET DEFNUM(B) = 5 ''MOVING'' CALL HIDER(B)
22             LET FLD.INT.DIST(B) = SAMIN.FLD(FLD)-50.0
23             LET FLD.FORM(B) = INT.F(P7.FLD(FLD))
24             IF BLADE.COND(B) EQ 2
25                LET FSPEED.FAC(B) = (P8.FLD(FLD)+1)/4
26                LET BLADE.COND(B) = 1
27             ALWAYS
28             IF SYS.TYPE(B) EQ 1 AND
                  (WPN.TYPE(B) EQ 4 OR WPN.TYPE(B) EQ 5)
29                LET FSPEED.FAC(B) = (P8.FLD(FLD)+1)/4
30             ALWAYS
31             LET TRAF.CONTROL(OBS.NUM,ROUTE(B),4) = 0.0
32             RETURN
33         OTHERWISE
34      ALWAYS
35      CALL JUNK.ALIVE GIVEN B YIELDING
                                  EQUIP.STAT, BRIDGE.STAT
36      IF BRIDGE.STAT NE 1 ''NO BRIDGING EQUIPMENT
37          IF TYP.FLD(FLD) EQ 6 OR EQUIP.STAT NE 1
38             IF TRAF.CONTROL(OBS.NUM,ROUTE(B),2) = 0.0
               ''NO BYPASS
39               SCHEDULE A STAND.TO GIVEN B NOW
40               LET DEFNUM(B) = 5 ''MOVE'' CALL HIDER(B)
41               LET FLD.INT.DIST(B) =
                                  SAMIN.FLD(FLD) - 50.0
42               LET TRAF.CONTROL(OBS.NUM,ROUTE(B),4) = 0
43               LET COUNTER = INT.F(SAMIN.FLD(FLD)/
44                (.5*LIN.SPDS(SYS.TYPE(B),WPN.TYPE(B),4))
                                                 /INTERV)
45               SCHEDULE A QUIK.MOVE(B,COUNTER,INTERV)
                                                    NOW
46             ELSE  ''A BYPASS EXISTS/B.B MUST HAVE ONE
47               LET COUNTER = 10   ''ENOUGH GET TURNED
48               SCHEDULE A TURN.AROUND GIVEN B NOW
49               SCHEDULE A QUIK.MOVE(B,COUNTER,INTERV)
                                                    NOW
50               SCHEDULE A STAND.TO GIVEN B IN 3 UNITS
51               FOR EVERY TNK IN PLT.UNIT(PLT(B))
52                                WITH SYS.TYPE(TNK) NE 3, DO
53                IF FLD.NO(TNK) NE FLD
54                SCHEDULE A TURN.AROUND GIVEN TNK NOW
55                SCHEDULE A QUIK.MOVE(TNK,COUNTER,INTERV)
                                                    NOW
```

```
56                ALWAYS
57              LOOP
58              LET FLD.INT.DIST(B) = RINF.C
59            ALWAYS
60            RETURN
61          OTHERWISE
62      ALWAYS
63      SCHEDULE A HEAVY.JUNK GIVEN B IN
                                          SIT.REP.TIME UNITS
64  ALWAYS
65  RETURN
66  END
```

27. Event GAP.JOCK

Purpose

Event GAP.JOCK gives entities that do not have a bypass, the ability to cover by direct fire a breach already in progress on their route of movement. This event is scheduled from routine GAP.ENTRY and event GAP.JOCK.

Given Argument Integer

B - The pointer to the entity that is trying to support another platoon's breach.

Events Scheduled

STAND.TO (this appendix)

QUIK.MOVE (this appendix)

Global Variable Real

TRAF.CONTROL (3-D) (Appendix A)

LIM.SPDS (3-D)

Indicates the limiting speeds for vehicles in a mounted attack.

Permanent Attribute Integer

F.PLT.UNIT

The first member of the set owned by the permanent entity, PLATOON LEADER.

Recursive Variable Integer

COUNTER

Contains the number of five second move increments that will take place for this entity. This movement update is accomplished by scheduling event QUIK.MOVE.

BRIDGE.STAT

Contains the status of the AVLBs in the platoon.

EQUIP.STAT

Contains the status of the blade tanks in the platoon.

FLD- Pointer to the field in question.

OBS.NUM

Receives the value of the field attribute NAM.FLD. This is the sequential id number in order of field creation.

TNK

Pointer to the first element in the platoon of the activating entity.

Recursive Variable Real

INTERV

A constant term that is set so that element locations are updated every five seconds during a small portion of the simulation.

SIT.REP.TIME

This is the time between status checks on the breaching operation.

Routines Called

HIDER

A Routine used to determine the micro-terrain elevation for a selected element.

JUNK.ALIVE (this appendix)

Temporary Attribute Integer

BAGGED.BOY

This variable is an argument for event STAND.TO. It carries the number of the entity to be moved.

DEFNUM

The current position or activity of an element.

1	full defilade
2	turret defilade
3	firing defilade
4	half vehicle defilade
5	moving (defilade determined by the terrain model)
6	reached final area in movement

FLD.NO

Name of the field involved in any pending internal action.

KKILL

Indicates whether an entity has sustained a catastrophic kill.

0	no
1	yes

MKILL

Indicates whether an entity has sustained a mobility kill.

0	no
1	yes

NAM.FLD

This is the sequential id number in order of field creation.

NUM.POINTER

This variable is an argument for event QUIK.MOVE. It carries the number of the entity to be moved.

REP.NUMBER

This variable is an argument for event QUIK.MOVE. It carries the number of iterations of scheduling left for this event to perform.

ROUTE

Indicates the number of the route along which the element is travelling.

0 not using a route

SABOT.SHOOTER

This variable is an argument for event GAP.JOCK. It carries the number of the activating entity.

SYS.TYPE

This represents the general class of the system of the entity.

1 tanks
2 mounted infantry
3 dismounted infantry
4 artillery
5 air
6 air defense
7 bunker
8 comm/ew/acq/intel
9 other

WPN.TYPE

Describes the specific system within the system code.

Temporary Attribute Real

FLD.INT.DIST

This attribute of an entity contains the distance to a pending field internal action.

INT.TIME

This variable is an argument for event QUIK.MOVE. It carries the time interval number when the next event is to be performed.

SAMIN.FLD

This field attribute is the semi-minor axis length of the elliptical field in meters.

Brief Explanation

Lines 9-16 When a breaching operation is ongoing by another platoon on this route, then a status check is made on this operation. If it is still in progress, another similar status update is scheduled to occur in the future.

Lines 17-22 The flag that indicates an in-progress breaching operation can be lifted/nullified by a successful breach or by a fatal attempt. In any case, the entity starts to move toward the gap obstacle. Routine

GAP.INTERNAL sorts out the appropriate actions to occur next.

CODE

```
 1  EVENT GAP.JOCK GIVEN B
 2  DEFINE B,OBS.NUM,FLD,TNK,EQUIP.STAT,BRIDGE.STAT,
          COUNTER AS INTEGER VARIABLES
 3  DEFINE SIT.REP.TIME, INTERV AS REAL VARIABLES
 4  LET FLD = FLD.NO(B)
 5  LET OBS.NUM = NAM.FLD(FLD)
 6  LET INTERV = 5.0  ''MOVE UPDATES
 7  LET SIT.REP.TIME = 15.0  ''TIME TO UPDATE STATUS
 8  IF KKILL(B) NE 1 AND MKILL(B) NE 1
 9     IF TRAF.CONTROL(OBS.NUM,ROUTE(B),4) NE 0.0
10        LET TNK =
   F.PLT.UNIT(INT.F(TRAF.CONTROL(OBS.NUM,ROUTE(B),4)))
11        CALL JUNK.ALIVE GIVEN TNK YIELDING
                                  EQUIP.STAT, BRIDGE.STAT
12        IF EQUIP.STAT = 1 OR BRIDGE.STAT = 1
13           SCHEDULE A GAP.JOCK GIVEN B IN
                                       SIT.REP.TIME UNITS
14           RETURN
15        OTHERWISE
16     ALWAYS
17        LET COUNTER = INT.F((SAMIN.FLD(FLD)*2.0)/
18          (.5*LIM.SPDS(SYS.TYPE(B),WPN.TYPE(B),4))
                                                 /INTERV)
19        SCHEDULE A QUIK.MOVE(B,COUNTER,INTERV) NOW
20        SCHEDULE A STAND.TO GIVEN B NOW
21        LET DEFNUM(B) = 5 ''MOVING'' CALL HIDER(B)
22        LET FLD.INT.DIST(B) = SAMIN.FLD(FLD) - 50.0
23  ALWAYS
24  RETURN
25  END
```

APPENDIX D

USER INPUT INSTRUCTIONS FOR THE STAR ENGINEER MODULE

A. INTRODUCTION

In order to emplace obstacles in a STAR Combat Simulation the obstacles must be planned in advance. Once the user has planned the use of obstacles and their effects on the tactical play of a simulation, then the input values to implement the plans in STAR may be determined. This Appendix consists of instructions for the user to input the required field parameters in order to represent obstacles in STAR. In addition, the necessary parameters for the representation of certain breaching equipment and different tactical options for the commander are also provided. In order to make this very clear, sample input values of the required data for the Engineer Effects Module are also included.

The first step is to determine the X and Y coordinates of the center of each obstacle. Once these values are known, then the size of the obstacle can be approximated using an ellipse. This is possible because an ellipse has a major and a minor axis. The routes of movement of entities

in the simulation may pass through some of the obstacle ellipses. Once the obstacles and routes have been planned, the user is ready to implement these plans in a STAR simulation.

B. OBSTACLE/FIELD REPRESENTATION

The first step in simulating obstacles in STAR is to input the OB.NUM (the number of obstacles in the scenario) and the RT.OB.NUM (the number of routes affected by obstacles). These input parameters are discussed in detail in Appendix A. The next step in creating obstacles on the STAR battlefield entails using the following three outlines of obstacle field attributes. These inputs are placed right after the OB.NUM and RT.OB.NUM numbers.

1. Minefields

The following define the attributes of the field used to simulate a minefield in STAR:

XC.FLD = X coordinate of ellipse center.

YC.FLD = Y coordinate of ellipse center.

SAMAJ.FLD = Length of the semi-major axis of ellipse in meters.

SAMIN.FLD = Length of the semi-minor axis of ellipse in meters.

ANGLE.FLD = Angle of semi-major axis measured

counterclockwise from the east in

degrees.

TYPE.FLD = 2

P1.FLD = 0(not active) or the number of mines in the

scatterable minefield.

P1.FLD = 0 or 1 to signify not activated or activated for

a patterned minefield.

P2.FLD = Trip wire length in meters for Scatterable

Anti-Personnel mines.

P2.FLD = Belt separation in meters for Patterned Anti-Tank

mines.

P3.FLD = mine type (1-11)

1 - M70 Scatterable Anti-Tank mine
2 - M56 Scatterable Anti-Tank mine
3 - M21 Hand-Emplaced Anti-Tank mine
4 - M15 Hand-Emplaced Anti-Tank mine
5 - M19 Hand-Emplaced Anti-Tank mine
6 - M14 Anti-Personnel Blast mine
7 - M25 Anti-Personnel Blast mine
8 - M16 Anti-Personnel Frag mine
9 - ADAM Anti-Personnel Frag mine
10 - M74 Anti-Personnel Frag mine
11 - Claymore Anti-Personnel Frag mine

P4.FLD = Probability of detection and avoidance.

P5.FLD = Probability of pushing a dead plow tank

(1-P5.FLD) = offset probability

P6.FLD = Knowledge level of this minefield

0 - No knowledge exists
2 - Knowledge exists
3 - Knowledge exists and a lane exists
(its location is known)

P7.FLD = The lane formation

(This and the offset formations are the last formations to appear in the appropriate part input deck.)

P8.FLD = Lane speed factor

(value between 0-1/small is slower)

P9.FLD = Plow activation delay time in seconds

P10.FLD = Plow speed factor

(value between 0-1/should be < P8.FLD)

P11.FLD = Push speed factor

(value between 0-1/should be < P10.FLD)

P12.FLD = Minefield detection distance

a. Input Example for a Patterned Minefield

This is the input data for a patterned/belted minefield.

53000 101500 500 200 90 2 1 50 3 .1 .5 0 3 .8 15 .6 .4 100

The location of the minefield center is:

X coordinate 53000

Y coordinate 101500

The semi-major axis length is 500 meters.

The semi-minor axis length is 200 meters.

The orientation angle of the minefield measured counterclockwise from the east is 90 degrees.

A minefield is type number 2.

The minefield is active, signified by the number 1.

The spread between mine belts is 50 meters.

This minefield was emplaced using M-21 mines (number 3).

The probability of detecting and avoiding a mine is .10 .
(tilt rods may be visually detected)

The probability that a tank will push a dead plow tank is .50.

The attacker does not have intelligence about this minefield. (knowledge level is 0)

The lane formation is the third formation in the formation input list for this example.

When a vehicle is travelling on a lane in a minefield, its speed is reduced by 20%. (input number is .8)

It takes 15 seconds for the plow tank to activate its plow.

When a vehicle is plowing, its speed is reduced by 40%. (input number is .6)

When a vehicle is pushing a dead plow tank, its speed is reduced by 60%. (input number is .4)

The distance travelled in a field before visual identification of the minefield by a vehicle is 100 meters.

b. Input Example for a Scatterable Minefield

This is the input data for a scatterable anti-tank minefield.

53000 101500 500 200 90 2 900 0 1 .4 .5 0 3 .8 15 .6 .4 30

The location of the minefield center is:

X coordinate 53000

Y coordinate 101500

The semi-major axis length is 500 meters.

The semi-minor axis length is 200 meters.

The orientation angle of the minefield from east is 90 degrees.

A minefield is type number 2.

The minefield is active and has 900 mines in it.

This type mine (anti-tank) does not have trip wires. (input number is 0)

This minefield was delivered using M-70 mines. (The input number is 1.)

The probability of detecting and avoiding a mine is .40.

The probability that a tank will push a dead plow tank is .50.

The attacker does not have intelligence about this minefield. (The knowledge level is 0.)

The lane formation is the third formation in the

formation input deck.

When a vehicle is travelling on a lane in a minefield, its speed is reduced by 20%. (input number is .8)

It takes 15 seconds for the plow tank to activate its plow.

When a vehicle is plowing, its speed is reduced by 40%. (the input number is .6)

When a vehicle is pushing a dead plow tank, its speed is reduced by 60%. (input number is .4)

The distance travelled in a field before visual identification of the minefield by a vehicle is 30 meters.

2. Gap Obstacles

The following defines the attributes on the field used to simulate a road crater, tank ditch or blown bridge (short-wet gap) in STAR:

XC.FLD = X coordinate of ellipse center.

YC.FLD = Y coordinate of ellipse center.

SAMAJ.FLD = Length of the semi-major axis of ellipse in meters.

SAMIN.FLD = Length of the semi-minor axis of ellipse in meters.

ANGLE.FLD = Angle of semi-major axis measured

counterclockwise from the east in

degrees.

TYPE.FLD = 4(tank ditch); 5(road crater); 6(blown bridge).

P1.FLD = The activation level of the obstacle.

0 - not activated
1 - activated

P2.FLD = 0

P3.FLD = 0

P4.FLD = 0

P5.FLD = 0

P6.FLD = Knowledge level of this obstacle.

0 - No knowledge exists
2 - Knowledge exists
3 - Knowledge exists and a lane exists
(we know where it is)

P7.FLD = The lane formation (This can be just a column

formation. There is no choice as in the

minefield case.)

P8.FLD = Lane speed factor.(A value between 0 and 1,

smaller is slower.)

P9.FLD = AVLB activation delay time in seconds.

(Typical values range from 180 to 300 seconds)

P10.FLD = Tank dozer breach time in seconds.

(Typical values range from 360 to 480 seconds)

P11.FLD = Tank self breach time in seconds.

(Typical values range from 400 to 600 seconds)

P12.FLD = Probability of a self breach getting stuck in the ditch.

a. Input Example for a Tank Ditch

This is the input data for a Tank Ditch.

53000 101500 500 200 90 4 1 0 0 0 0 0 3 .8 180 360 400 .3

The location of the tank ditch center is:

X coordinate 53000

Y coordinate 101500

The semi-major axis length is 500 meters.

The semi-minor axis length is 200 meters.

The orientation angle of the tank ditch from east is 90 degrees.

A tank ditch is type number 4.

The tank ditch is an actual/active obstacle signified by the number 1.

The next four entries are zeros.

The attacker does not have intelligence about this tank ditch. (The knowledge level is 0.)

The lane formation is the third formation in the formation input list.

When a vehicle is traveling on a lane through/over a gap obstacle, its speed is reduced 20%. (input number is .8).

It takes 180 seconds for the AVLB to bridge the gap in this example.

It takes 360 seconds for the blade tank to breach this gap.

It takes 400 seconds for a self breach to take place.

The probability that the vehicle will be immobilized during a self breach is .30 .

3. Decision Ellipses

The following defines the attributes on the field used to simulate the decision ellipses for all types of obstacles in STAR: (WARNING: A blown bridge obstacle must always have a bypass route and a bypass movement control point set. The vehicles cannot "bull through" an obstacle of this type.)

XC.FLD = X coordinate of ellipse center.

YC.FLD = Y coordinate of ellipse center.

SAMAJ.FLD = Length of the semi-major axis of ellipse in meters. (This must be large enough so that each element in a platoon hits the decision ellipse.) (CAUTION: Make sure routes are not too close to each other. The user cannot let units on one route hit a decision ellipse on

a different route, except when lateral movement is being performed.)

SAMIN.FLD = Length of the semi-minor axis of ellipse in meters.

ANGLE.FLD = Angle of semi-major axis from east in degrees.

TYPE.FLD = 3 (minefields) and 7 (gaps).

P1.FLD = The field attribute that allows the tactical option to bypass an obstacle if the battlefield situation warrants it.

0 - This route has no bypass available, the entity will move to the obstacle and attempt a self-breach. ("bull through")

route number - This is the bypass route to be used.

P2.FLD = The bypass MCP that is on the bypass route. (if not applicable enter a 0)

P3.FLD = The obstacle number in front of this ellipse. This allows the decision ellipse to check on the level of knowledge about the existence of the obstacle.

0 - This is a decision ellipse on a route that circumvents the obstacle. The only purpose of this decision ellipse is to get lateral

moving entities back into route formations.

obstacle number - This is the sequential input number of the obstacle to be encountered in front of this decision ellipse.

P4.FLD = 0 (set to present route during simulation)

P5.FLD = 0 (set to present MCP during simulation)

P6.FLD = 0 (set to a counter during simulation)

P7.FLD = 0 (not used)

P8.FLD = 0 (not used)

P9.FLD = 0 (not used)

P10.FLD = 0 (not used)

P11.FLD = 0 (not used)

P12.FLD = 0 (not used)

a. Input Example for a Minefield Decision Ellipse

This is the input data for a Minefield Decision Ellipse.

53700 101700 100 100 90 3 14 3 1 0 0 0 0 0 0 0 0 0

The location of the Decision Ellipse center is:

X coordinate 53000

Y coordinate 101500

The semi-major axis length is 100 meters.

The semi-minor axis length is 100 meters.

The orientation angle of this decision ellipse from the

east is 90 degrees.

A minefield decision ellipse is type number 3.

The bypass route is number 14.

The bypass MCP is 3.

The obstacle in front of this decision ellipse is field number 1.

The next 9 entries are zeros.

C. ARMORED VEHICLE LAUNCHED BRIDGE REPRESENTATION

The target dimensions for the vehicle launched bridges were found by finding the dimensions of the bridges on the US AVLB and the Soviet MTU-20 and putting these bridge types on the M-1 and T-72 tanks, respectfully.

The next line is the US M-1 AVLB.

1 4 1.5 2.24 .0 3.74 4.0 3.7 8.0 11.15 .0 .60

The next line is the Soviet T-72 bridge.

1 5 1.35 1.92 .0 3.27 3.3 3.38 5.2 11.6 .0 .59

These entries are specified as follows:

1. system type
2. weapon type
3. hull height (meters)
4. turret/bridge height (meters)
5. not applicable
6. entire vehicle height (meters)
7. bridge width (meters)
8. frontal hull width (meters)
9. flank hull width (meters)
10. unlaunched/carried bridge width (meters)
11. not applicable
12. the percent of total vehicle height above hull defilade.

The above dimensions are for future systems. If today's arsenal is to be represented, then actual vehicle configurations and dimensions must be used.

Limiting speeds for these bridges have to be added to the input deck along with the dismounted limiting speeds.

Bridges neither carry nor shoot ammunition. This characteristic can be simulated by putting a zero in the types of ammunition block on input. An example follows:

```
1   4   0
1   5   0
```

The first entry is the system type.

The second entry is the weapon type.

The third entry specifies the different ammunition types for this vehicle.

The array POINT.HOLD should be modified as follows:

```
1 4       9 1 0 0 0 0 0 0 0 0 0 0 0 0 0
```

The system/weapon target selection array for the vehicle launched bridges have to be added so that they are unable to select targets for firing.

```
1 0 0 0 0      (0-1000 meters)
1 0 0 0 0      (1000-2000 meters)
1 0 0 0 0      ( > 2000 meters)
```

The same additions have to be made for the T-72 bridge.

```
1 5       9 1 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
1 0 0 0 0
1 0 0 0 0
1 0 0 0 0
```

These new entities, Armored Vehicle Launched Bridges, must also be added to the target selection arrays of their respective enemy entities.

The suppression data for each type bridge must be added to the input deck.

D. OFFSET FORMATIONS

The next area of input that must be discussed is the lane and offset formations. These new formations allow the Engineer Module to simulate lanes, breaching operations, and the detour of elements around killed vehicles. The following example is a sample of a FORM.OFFSET array.

```
7
1 5 0 0 0 -50 0 50 0 -100 0 100
2 5 0 0 50 0 -50 0 100 0 -100 0
3 5 0 0 -50 0 -100 0 -150 0 -200 0
4 5 0 10 -50 10 -100 10 -150 10 -200 10
5 5 0 20 -50 20 -100 20 -150 20 -200 20
6 5 0 30 -50 30 -100 30 -150 30 -200 30
7 5 0 40 -50 40 -100 40 -150 40 -200 40
```

The first number (7) indicates that there are seven formations to follow.

The first formation in the list is a five place line formation.

column 1 Specifies this formation as number one for the program.

column 2 Specifies that there are five postions in this formation.

column 3-4 Specify the X and Y offsets for the vehicle. This entity has a formation postion equal to one.

Each succeeding pair of numbers specifies the formation offsets for vehicles with formation positions which are in increments of one. A vehicles formation position is the same as his POS.IN.PLT.AREA .

The second formation is a five place column formation with tank one (0,0), in the middle of the platoon.

The third formation is a five place breaching formation with the first tank in the lead. This tank should be the plow tank for this platoon. Tank one can be the plow tank if the element data is assigned in the proper manner.

The fourth formation is the first offset formation. This formation would be the first used if a dead vehicle became an obstacle to movement on the route. All elements are in the same relative positions within their platoon as dictated by formation number three, the breach formation, except that the platoon has been displaced 10 meters to the left of its route.

The fifth, sixth, seventh formations are the same as formation number three, except offsets of 20, 30, 40 meters, respectively, have been added as Y-displacements off the route. The number of offset formations, not including the lane/breach formation, depends on the number of mechanized entities in the largest platoon in the scenario. The number of offset formations should be at least this large.

E. TANK BREACHING ATTACHMENTS

The Mine Plow and Tank Dozer Blade characteristics for individual entities are submmited for the simulation as part of the element data in the input deck. The input position definitions are defined below.

position:

1	NAME	The element number of the entity.
2	SYS.TYPE	This represents the general class of the system of the entity. 1 Tanks 2 Mounted infantry 3 Dismounted infantry 4 Artillery 5 Air 6 Air defense 7 Bunker 8 Comm/ew/acq/intel 9 Other
3	WPN.TYPE	Describes the specific system within the system code. For example, system type is a tank and weapon type of seven

		for this system indicates a Soviet T-72 tank.
4	SEC	Indicates the number of the SECTION to which the entity belongs.
5	PLT	The number of the PLATOON to which the entity belongs.
6	CO	The number of the COMPANY to which the entity belongs.
7	BN	Indicates the number of the BATTALION to which the entity belongs.
8	SQDVEH	Indicates the number of the entity's squad vehicle.
9	PLTLDR	The element number of the entity's PLATOON LEADER. The PLATOON LEADER must be the first element input in his PLATOON.
10	COCDR	The element number of the entity's COMPANY COMMANDER. The COMPANY COMMANDER is the first entity input in his COMPANY.
11	START.AREA	The number which identifies the starting movement area for this entity.
12	ALIVE.DEAD	Indicates whether the entity is alive

or dead.

0 Alive
1 Dead
2 Alive mounted in carrier.

13 FIR.MODE Indicates the mode of fire that a dismounted element is to use.

0 Fires in a semi-automatic mode.
1 Fires in an automatic mode.

14 PLOW.COND This unit attribute gives the tank a mine plow.

0 No plow
1 Plow attached

15 BLADE.COND This unit attribute gives the tank a bull dozer blade.

0 No blade
1 Dozer blade attached

The following is an example of two typical element data entries:

45 1 7 45 15 4 2 45 45 40 601 0 0 1 0

46 1 7 46 15 4 2 46 45 40 601 0 0 0 1

In this example - Element number 45 is a Soviet T-72 tank with a mine plow attachment.

- Element number 46 is a Soviet T-72 tank with a dozer blade attachment.

In this appendix, the authors have not tried to give

complete user input instruction for STAR. The input instructions given are enough for the current user of the STAR model to run the model with the Engineer Effects Module. If more complete instructions are required, then the reader is directed to the STAR Combat Model documentation currently in use.

LIST OF REFERENCES

1. Department of the Army, U. S. Army Tradoc Systems Analysis Activity, UNCLASSIFIED Transmittal letter, SUBJECT: History of Mine Warfare, TM 35-80, 4 February 1981.

2. Department of the Army, HQ, VII Corps UNCLASSIFIED Letter, SUBJECT: Summary Report of the 563D Engineer Bn Anti-Tank Ditch Program, 5 July 1978.

3. Hartman, J.K., The STAR Field Module, Naval Postgraduate School, Technical Report NPS 55-80-023, June 1980.

4. Slattery, P.J., A Structure for the Development of an Engineer Module, M.S. Thesis, Naval Postgraduate School, Monterey, CA, December 1980.

5. Gibson, L.P., A Method to Determine Divisional Engineer Battalion Training Measures of Effectiveness, M.S. Thesis, Naval Postgraduate School, Monterey, CA, September 1978.

6. Department of the Army, FM 5-100, Engineer Combat Operations, March 1979.

7. Department of the Army, FM 90-7, Obstacles, December 1977.

8. Department of the Army, TC 7-24, Antiarmor Tactics and Techniques for Mechanized Infantry, September 1975.

9. Department of the Army, FM 20-32, Mine/Countermine Operations at the Company Level, November 1976.

10. Department of the Army, FM 5-15, Field Fortifications, June 1972.

11. Department of the Army, FM 5-34, Engineer Field Data, September 1976.

12. Department of the Army, FM 71-2, The Tank and Mechanized Infantry Battalion Task Force, June 1977.

13. Donnelly, C.N., "Combat Engineers of the Soviet Army" International Defense Review, v. 2, April 1978.

14. Honeywell, Countermine Warfare Analysis Final Report,

Mission Analysis Group, Honeywell Defense Systems Division, June 1981.

15. Department of the Army, U. S. Army Engineer Center and School, Ft Belvoir, VA., Combat Engineer Systems Handbook, 8-9 April 1981.

16. Department of the Army, Combined Arms Combat Development Activity, Threats Office, HB 550-2, Organization and Equipment of the Soviet Army, 15 July 1980

17. Department of the Army, U. S. Army Intelligence and Threat Analysis Center, Soviet Army Operations, April 1978.

18. Department of the Army, U. S. Army Engineer Center and School, Department of Combined Arms, Ft Belvoir, VA., Opposing Forces, 1977.

19. Donnelly,C.N.,"Soviet Tactics for Overcoming NATO Anti-tank Defenses" Parts 1 and 2, International Defense Review, Sept/Oct 1980 and Nov/Dec 1980.

20. The BDM Corporation, A Combat Simulation for Evaluating the Worth of Military Equipment and Tactics in Company and Battalion Level Armor/Mechanized Infantry Warfare, v. 2, June 1981.

21. Scientific Control Systems Ltd., Minefields and Barrier Combat Simulation, September 1976.

22. Department of the Army, U. S. Army Tradoc Systems Analysis Activity, UNCLASSIFIED Letter, Subject: Combined Arms and Support Task Force Evaluation Model: A Brief Description, May 1981.

23. USACDC Report, Army Small Arms Requirements Study (ASARS) Battle Model, v 2, 1972.

24. Department of the Army, Engineer Center, Combat Developments Experimentation Command, Unclassified Draft Report, SUBJECT: Detection and Avoidance of Scatterable Mines, May 1981.

25. Buckling B.L., and others, Camp Drum Test of Mine Effectiveness, Picatinny Arsenal,Dover, New Jersey, Technical Report PA-TM-2067, December 1972.

26. Department of the Army, Engineer Center, Combat Developments Experimentation Command, Unclassified Final Report, SUBJECT: Tactical Effectiveness of Minefields in the Anti-Armor Weapons Systems (TEMAWS), June 1977.

27. Hartman J. K., Ground Movement Modelling in the STAR Combat Model, Naval Postgraduate School, Technical Report NPS55-80-021, May 1980.

28. Department of the Army, Engineer School, Ft Belvoir, VA., Unclassified Letter, Formula for Mine Probability of Kill., 8 July 1981.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 55 Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
4.	Chief TRADOC Research Element Monterey Naval Postgraduate School Monterey, California 93940	10
5.	Associate Professor James K. Hartman Code 55Hh Department of Operations Research Naval Postgraduate School Monterey, California 93940	5
6.	Associate Professor S. H. Parry Code 55Py Department of Operations Research Naval Postgraduate School Monterey, California 93940	5
7.	Headquarters US Army Training and Doctrine Command ATTN: Director, Studies and Analysis Directorate, Mr. S. Goldberg Fort Monroe, Virginia 23651	1
8.	Associate Professor A. L. Schoenstadt Code 53Zh Department of Mathematics Naval Postgraduate School Monterey, California 93940	1
9.	Professor James G. Taylor Code 55Tw Department of Operations Research Naval Postgraduate School Monterey, California 93940	1

10. Office of the Commanding General 1
US Army Training and Doctrine Command
ATTN: General Glenn Otis
Fort Monroe, Virginia 23651

11. Headquarters 1
US Army Training and Doctrine Command
ATTN: ATCG-T (BG Morelli)
Fort Monroe, Virginia 23651

12. Office of the Commanding General 1
US Readiness Command
ATTN: General Donn A Starry
MacDill AFB, Florida 33621

13. Deputy Under Secretary of the Army 1
for Operations Research
Room 2E261, Pentagon
Washington, DC 20310

14. LTG Howard Stone 1
Commanding General
US Army Combined Arms Center
Fort Leavenworth, Kansas 66027

15. Director 1
Combined Arms Combat Development Activity
ATTN: ATZL-CAC-A (Mr. Lee Pleger)
Fort Leavenworth, Kansas 66027

16. Director 1
Combat Analysis Office
ATTN: Mr. Kent Pickett
US Army Combined Arms Center
Fort Leavenworth, Kansas 66027

17. Command and General Staff College 1
ATTN: Education Advisor
Room 123, Bell Hall
Fort Leavenworth, Kansas 66027

18. Dr. Wilbur Payne, Director 1
US Army TRADOC Systems Analysis Activity
White Sands Missile Range, New Mexico 88002

19. Headquarters, Department of the Army 1
Office of the Deputy Chief of Staff for
Operations and Plans
ATTN: DAMO-2D
Washington, D.C. 20310

20. Commander 1
US Army Concepts and Analysis Agency
ATTN: MOCA-WG
8120 Woodmont Avenue
Bethesda, Maryland 20014

21. Director — 1
US Army Material Systems Analysis Activity
ATTN: DRXSY-CM (Mr. Bill Niemeyer)
Aberdeen Proving Grounds, Maryland 21005

22. Director — 1
US Army Night Vision and Electro-Optical Lab
ATTN: DEL-NV-VI (Mr. Frank Shields)
Fort Belvoir, Virginia 22060

23. Director — 1
USATRASANA
ATTN: Mr. Ray Heath
White Sands Missile Range, New Mexico 88002

24. Director — 3
Combat Developments, Studies Division
ATTN: MAJ W. Scott Wallace
US Army Armor Agency
Fort Knox, Kentucky 40121

25. Commandant — 1
US Army Field Artillery School
ATTN: ATSA-CD-DSWS
Fort Sill, Oklahoma 73503

26. Director — 1
Combat Developments
US Army Aviation Agency
Fort Rucker, Alabama 36362

27. Director — 1
Combat Developments
US Army Infantry School
Fort Benning, Georgia 31905

28. Director — 5
Combat Developments
ATTN: ATZA-CDE (CPT James Mudd)
US Army Engineer School
Fort Belvoir, Virginia 22060

29. Director — 1
Combat Developments
ATTN: ATSA-CDP-S
US Army Air Defense Agency
Fort Bliss, Texas 79905

30. Commander — 1
US Army Logistics Center
ATTN: ATCL-OS (Mr. Cameron/CPT McGrann)
Fort Lee, Virginia 23801

31. Commandant
US Army Signal School
ATTN: LTC Harnagal
Fort Gordon, Georgia 30905 — 1

32. CPT James V. Mudd
35 Heussy Ave.
Buffalo, New York 14220 — 1

33. CPT Stephen C. Main
US Readiness Command
ATTN: ASG
MacDill AFB, Florida 33621 — 1

END

DATE
FILMED

10 82

DTIC